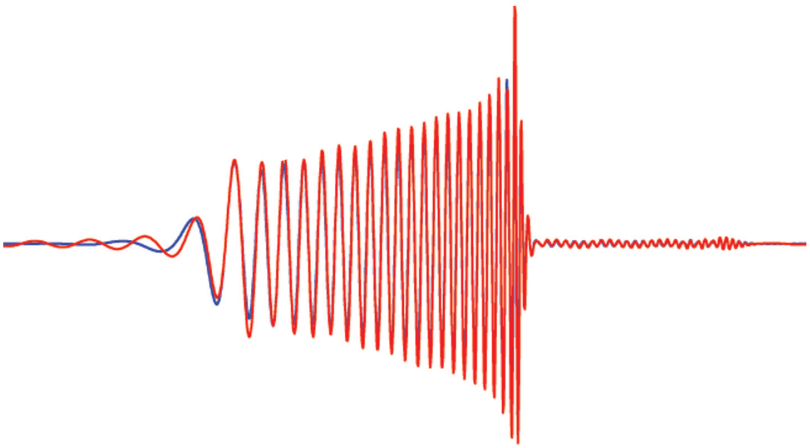
COMPUTATIONAL INFRASTRUCTURE FOR GEODYNAMICS (CIG)

Mineos

User Manual
Version 1.0



Guy Masters
Misha Barmine
Susan Kientz

www.geodynamics.org

Mineos

Guy Masters
Version 1.0

March 19, 2007

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Mineos Package Overview

The **Mineos Package** by Guy Masters consists of four programs: **minos_bran**, **eigcon**, **green**, and **syndat**. Functionally, the programs break into two subgroups. The first subgroup contains the programs that produce the normal eigenfunctions and eigenfrequencies, **minos_bran** and **eigcon**. Information flow into and out of these two programs, which we call the eigenfunction system, is summarized in Figure 1.1. The second subgroup, referred to as the synthetic seismogram system, comprises the programs that read the output from **eigcon** and compute Green's functions and synthetic seismograms. These programs are **green** and **syndat**, and information flow through these programs is summarized in Figure 1.2.

The four programs are executed in sequence. Some of the more important files that are produced along the way and at the end are formatted into an extension of the CSS-3.0 relational database schema. Each file is an ASCII flat file that can be read with a text editor, but has the advantage of also being subsumable into a database system such as ORACLE, Postgress, MySQL, or Antelope. This *multi-tiered access* is a design goal of the I/O system of the **Mineos** package. The potential disadvantage of using the relational database framework is that the files in the schema are formatted ASCII files that are better generated programmatically than by hand.

Under the CSS schema, the files are identified by a database name (*dbname*) and a suffix that specifies a particular type of file (or relation). An example would be *mineos.site*, which is a site table (or relation) for the database named *mineos*. To the many users of Antelope, some of these files will be transparent as they are part of the core CSS-3.0 definition. Examples include the *.site*, *.sitechan*, and *.wfdisc* relations. Other tables are extensions to CSS, such as the *.eigen* relation which contains parametric information for the eigenfunctions and points to a much larger direct access file containing the eigenfunctions. The package does not slavishly adhere to CSS, however. For example, event information is contained in a single flat file that summarizes the information that would be contained in the *.origin*, *.centryd,* and *.moment* relations in CSS. The input 1D model file also is not part of the CSS schema. File formats, therefore, are a hybrid with CSS and extensions to the CSS core.

The final output synthetic seismograms are represented by a *.wfdisc* relation, which points to binary waveform files. The synthetic, therefore, can be read and displayed by Antelope programs such as `dbpick` or `dbe`. The binary waveform files themselves, however, can be converted upon completion into the SAC format, with the SAC headers sufficiently populated so that the SAC program can be used to read in, manipulate, and display the waveforms. Such multi-tiered access is designed to facilitate user interaction with the synthetics.

1.2 Mineos History

The original algorithm was based on direct numerical integration of the governing differential equations (variable order, variable step-size Runge-Kutta, up to the eighth order) [9]. This initial version was coded by J. Freeman Gilbert circa 1966 while at the Institute of Geophysics and Planetary Physics (IGPP), Uni-

versity of California, San Diego. Tests showed the code, then called EOS, was superior to Burlisch-Stoer (despite claims to the contrary in *Numerical Recipes*). The variable step size was based on the growth rate approximated from estimates of the eigenvalues of the matrix A [7].

Finding the solution for spheroidal modes was computationally unstable, Gilbert and Backus suggested using the method of minors [6]. This was implemented in 1980 by John Woodhouse of Harvard University, who included a clever method of computing eigenfunctions and, later, a mode counter for spheroidals. Shortly thereafter Guy Masters of IGPP, UC San Diego, began working with the code, adding counters for toroidals and radials.

The codes were benchmarked against the Rayleigh-Ritz code of Ray Buland circa 1981. Slight differences discovered were tracked down to the use of a slightly different value of the gravitational constant. Around 1985, the code was modified to compute accurate eigenfunctions of "difficult" modes, e.g., Stoneley and IC modes, and the code was renamed Mineos.

The Mineos code has been uncopyrighted and handed down from investigator to investigator until donated to the community with an open source license via Computational Infrastructure for Geodynamics (CIG) in 2006 by Guy Masters and Michael Ritzwoller of University of Colorado at Boulder. At that time, the code was cleaned up and placed under source control, and its user documentation assembled and revised, by Misha Barmine (U. Colorado at Boulder). CIG released Mineos under a GNU free software license in March 2007.

CIG is making this source code available to you in the hope that the software will enhance your research in geophysics. CIG requests that in your oral presentations and in your papers that you indicate your use of this code and acknowledge the authors of the code and CIG (`geodynamics.org`).

1.3 Eigenfunction System of Programs

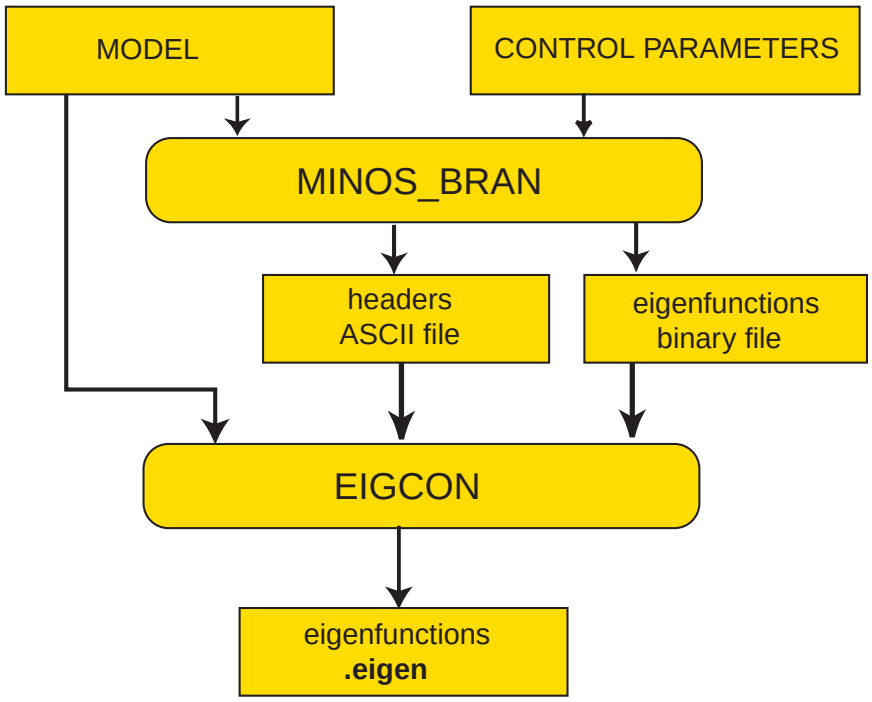


Figure 1.1: Summary of information flow through the eigenfunction system of programs comprising **minos_bran** and **eigcon**.

The program **minos_bran** is the work-horse of the **Mineos** Package. Given a 1D model of the Earth and the normal mode band of interest, i.e., defined in terms of a range of frequencies (*fmin, fmax*) and

normal mode indices (*nmin, nmax, lmin, lmax*), **minos_bran** computes and outputs the eigenfunctions and eigenfrequencies of the model for spheroidal, toroidal, or radial modes (optionally). The output from **minos_bran** is in an informally defined pair of files, one an ASCII file containing the input model and output normal mode parameters and the other a binary file containing the eigenfunctions from the free surface to the Earth's center. The size of the output depends on *fmax*, the highest desired frequency, and the number of radial knots in the input model file. If the desired frequencies extend to periods as short as 6 sec, the eigenfunctions and eigenfrequencies of more than 150,000 spheroidal and 100,000 toroidal modes will be computed if all dispersion branches are chosen.

The eigenfunctions themselves may be interesting to some users if sensitivity kernels, for examples, are desired. The normalization of the eigenfunctions that emerge from program **minos_bran** is discussed in Chapter 3.

The program **eigcon** repackages the eigenfunctions in two principal ways. First it renormalizes the eigenfunctions and then truncates the tabulation to extend only to a cut-off depth which is intended to be the depth of the deepest earthquake considered. This truncation greatly reduces the size of the eigenfunction file and speeds computation of the Green's functions. In addition, **eigcon** reformats the output eigenfunction file into a binary file that is much more independent of computer architecture than that which emerges from **minos_bran**.

The final output is represented more formally than the output from **minos_bran**, based on the *.eigen* relation, which points to the eigenfunction file on disk. The *.eigen* table is an extension of the CSS database schema.

1.3.1 Synthetic Seismogram System of Programs

Synthetic seismograms are produced in two stages. In the first stage, program **green** computes Green's functions and, in the second stage, program **syndat** transforms the Green's functions to synthetic seismograms by convolving them with a centroid moment tensor and input event half-rise time.

Program **green** computes the Green's function for an event at an input depth. The primary input is the *.eigen* relation(s) output from **eigcon**. Typically, there may only be two *.eigen* files input, one for spheroidal and another for toroidal modes. Radial modes could constitute another input file. However, the input is sufficiently flexible to allow the user to separate the input eigenfunctions into more files, if desired. It is up to the user to ensure that the files contain unique normal modes, however. The stations and channels for which Green's functions are produced by program **green** are driven by the input *.sitechan* table. The *.site* table is also needed by **green** to provide the station coordinates. The *.sitechan* table, therefore, must contain all and only those stations and channels desired by the user. The user must strictly adhere to the format of these files. A natural way to do this is to order a dataless (or data-full) SEED volume (for example, from the IRIS DMC) for the stations of interest, and run RDSEED or an Antelope product (e.g., sd2de, seed2db) to convert to the two CSS tables. Event coordinates and depth are contained in an unformatted, single lines file that we call *cmt_event*. This file must be created and input into program **green** and **syndat**.

The output of program **green** is a *.wfdisc* file (in the CSS-3.0 database schema) in which each row corresponds to a given station:channel pair and points to the Green's function on disk. Antelope products can be used to view the Green's functions in which each station:channel set of six waveforms is multiplexed into a single waveform.

Program **syndat** convolves the Green's functions pointed to by the *.wfdisc* relation that emerges from **green** with the centroid-moment tensor and half-rise time of the chosen event that is contained in file *cmt_event*. The output is a *.wfdisc* in which each row corresponds to a single station:channel pair and points to the associated waveform on disk presented in velocity ground units (nm/sec). Transfer functions to convert to instrument counts for direct comparison with data are not part of the package.

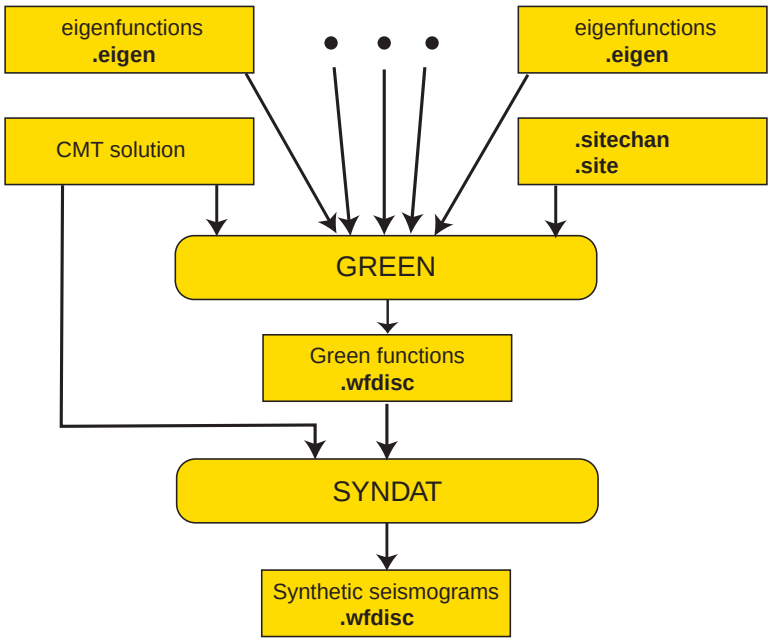


Figure 1.2: Summary of information flow through the synthetic seismogram system of programs comprising **green** and **syndat**.

Chapter 2

Installation and Getting Help

2.1 Getting Started

There are examples for running the four programs in Chapter 8. In addition, to help the user get started, the standard installation comes with several input files.

Model files. Four model files are presented: **prem_noocean.txt, prem_ocean.txt, NRussia.txt, CPacific.txt**. The first two are for PREM, one with an ocean and the other in which the ocean has been filled with solid crust. The other two model files are a continental and an ocean point, in N. Russia and the C. Pacific, respectively. The model in the latter two cases is the CU-Boulder 3D model in the top 400 km, a 3D model from Harvard through the rest of the mantle, underlain by PREM in the core, presented on a 2×2 degree grid worldwide.

Eigenfunction files. Output from **eigcon** is presented for the 1D model PREM without an ocean. There are spheroidal and toroidal files. For spheroidal modes, there is **prem_noocean_S.eigen** with the associated eigenfunction file that's contained in the directory **prem_noocean_S.eigen.dat**, simply called **eigen**. Similarly, for toroidal modes there is **prem_noocean_T.eigen** and directory **prem_noocean_T.eigen.dat**. Eigenfunctions are computed with the following characteristics: (fmin, fmax) = (0, 125.0 mHz), (nmin, nmax) = (1, 30), (lmin, lmax) = (1, 1631). This produces all toroidal and the vast majority of spheroidal modes up to an 8 sec period. There are about 28,000 spheroidal and 28,000 toroidal modes. Truncation depth of the eigenfunctions is 1000 km. The eigen files for the spheroidal and toroidal modes are about 52 and 22 MB, respectively.

Event files. There is a single event file called **china_cmt_event.txt** that contains Harvard CMT information for an event in Southern China.

Station:Channel files. Two sets of station:channel files are included: (1) **long.site** and **long.sitechan** and (2) **short.site** and **short.sitechan**. The "long" files are a long list of about 150 stations and the "short" list is about 15 stations at various distances from the event in S. China.

2.1.1 Utilities

Currently we provide four utilities: **cucss2sac**, **eigen2asc**, **endi**, **simpledit**, and **creat_origin**.

- **cucss2sac** converts synthetic waveforms represented in CSS3.0 (i.e., *.wfdisc* relation pointing to binary formatted waveforms on disk) into SAC or ASCII formatted waveform files.

  Note that the seismological community provides us with other CSS to SAC converters, and you may use them at your own risk. For example, for the SUN platform, the distribution set `css2sac-3.0.3.tar.z` is available from IRIS (`www.iris.edu`). The utility from that site, however, doesn't work under Linux or Windows platforms.

- **eigen2asc** prints out on the standard output device the requested eigenfunctions from *.eigen* relation (CSS3.0 extension). The eigenfunctions can be requested by order numbers (n, l) or by (n, T). In the second case, **eigen2asc** matches the order number l with the closest period to T. Output can be redirected to a file for further usage.
- **endi** swaps the order of word bytes in binary files. The length of a word is any integer number, for example, 2, 4, 6, 16, etc. **endi** can be used for BIG_ENDIAN-LOW_ENDIAN binary data conversion. Note that this program upgrades files in place.
- **simpledit** is a simple filter program. It converts a manually created (text editor) unformatted ASCII file with station and channel information into CSS3.0 *.site* and *.sitechan* relation tables.
- **creat_origin** is a shell script that converts an event file (e.g., **china_cmt_event.txt**) into the CSS3.0 *.origin* relation.

For more details, see Chapter 4.

2.1.2 System Requirements

- 32-bit processor (64-bit is recommended)
- Unix Operating System
- 512 MB of RAM
- FORTRAN 77 and C compilers. Currently, the **Mineos** package was tested under SunOS 5.7/f77/cc and under Linux RedHat/f77/g77.

  To check if you have these compilers, type at the command line:

  ```
  $ f77
  $ g77
  $ gcc
  $ cc
  ```

- At least 1 GB of available hard-disk space
- autoconf 2.59 or higher version and automake 1.8.4 or higher version. This requirement is only for software developers.

2.2 Download and Configure Mineos

1. Download the source package from the CIG website, or check-out the source from the SVN (Subversion) server

   - Go to CIG Software Packages: Mineos (`geodynamics.org/cig/software/packages/seismo/mineos`) and download the source package. Extract the source from the tar file:

     ```
     gunzip -c mineos-1.0.0.tar.gz | tar xf -
     cd mineos-1.0.0
     ```

   ~ OR ~

   - Check out the source from the Subversion server:

     ```
     svn co http://svn@geodynamics.org/cig/seismo/1D/mineos/trunk mineos
     cd mineos
     ```

2. If you checked-out the source using Subversion, run "autoreconf -i" to generate the 'configure' script and related files. (Skip this step if you downloaded the source package. The resulting files are included in the tar file.)

```
$ autoreconf -i
```

3. Run "configure"

```
./configure --prefix=$HOME/cig
```

4. Run "make"

```
make
```

5. Run "make install"

```
make install
```

6. Untar the DEMO tar file to access the provided input files

```
cd DEMO
gunzip -c DEMO.tar.gz | tar xf -
```

7. Now you can run using the demo input files. For example, under the Bash shell:

```
export PATH=$HOME/cig/bin:$PATH
cd DEMO3
./RUN_MINEOS.sh prem_noocean
```

2.3 Support

The primary point of support for Mineos is the CIG Computational Seismology Mailing List (`cig-seismo@geodynamics.org`). Feel free to send questions, comments, feature requests, and bugs to the list. The mailing list is archived at

```
cig-seismo Archives (geodynamics.org/pipermail/cig-seismo/)
```

You may also use the bug tracker

```
Roundup (geodynamics.org/roundup)
```

to submit bugs and requests for new features.

Chapter 3

Running the Mineos Programs

3.1 minos_bran Program

The **minos_bran** program produces the solution of the seismic normal mode eigenvalue-eigenfunction problem. The program evaluates the eigenvalues and eigenfunctions for radial, spheroidal (S) and toroidal (T) modes. The model of the Earth is self-gravitating, spherically symmetric, transversely isotropic, and attenuative. **minos_bran** is written in the FORTRAN-77 language.

3.1.1 Command Line

There are three different ways to start the **minos_bran** program:

1. Interactive dialog for setting input parameters:

   ```
   minos_bran
   ```

2. Single shell command with redirection of the standard input to the parameter file *parameter_file* containing exactly the same information and in the same order as an interactive dialog – one answer per line:

   ```
   minos_bran < parameter_file
   ```

3. Direct shell script. In this case, the contents of the parameter file are directly included into the shell script between the delimiters "WORD":

   ```
   minos_bran << WORD
   ........
   ........
   WORD
   ```

See Chapter 8 for examples.

The parameter file consists of six lines:

Line 1: *model_file*
model_file is the path to the 1D input model file. **Format:** Text string up to 256 characters long.

Line 2: *out_plain_file*
out_plain_file is the path to the output ASCII file. The file contains a model listing and a summary of mode properties. **Format:** Text string up to 256 characters long.

Line 3: *out_bin_file*
out_bin_file is the path to the output FORTRAN unformatted binary file. The file is a collection of eigenfunctions in a binary representation. The file name "none" is a special name; "none" will suppress the calculation of eigenfunctions. **Format:** Text string up to 256 characters long.

Line 4: *eps, wgrav*
The parameter *eps* controls the accuracy of the Runge-Kutta integration scheme. The relative accuracy of an eigenfrequency is a factor 2 to 3 times *eps*. Parameter *eps* also controls the precision with which a root is found and the minimum relative separation of two roots with the same angular order. It is safe to set $eps = 10^{-7}$ for periods greater than 10 seconds. For periods between 5 and 10 seconds, it has to be set to $10^{-12} - 10^{-10}$. *wgrav* is the frequency in millihertz (mHz) above which gravitational terms are neglected; this gives about a factor of 3 increase in speed. **Format:** Unformatted.

Line 5: *jcom*
jcom is the type of oscillation. $jcom = 1$ for radial modes, $= 2$ for toroidal modes, $= 3$ for spheroidal modes, and $= 4$ for inner core toroidal modes. **Format:** Unformatted.

Line 6: *lmin, lmax, wmin, wmax, nmin, nmax*
$lmin$, $lmax$ define the range of angular orders l to be computed. For radial modes, $jcom = 1$, $lmin$, $lmax$ are read in but are not used. $wmin$, $wmax$ define the frequency range to be computed (in millihertz). $nmin$, $nmax$ specify the range of dispersion branch numbers n to be computed. $n = 0$ is the fundamental mode. **Format:** Unformatted.

3.1.2 Input Data

model_file The model file is a plain ASCII file that may be given in either tabular or polynomial form. The first two lines in the model file are common. The rest of the file depends on the setting and file type. A more detailed description of the model file is:

Line 1: *title*
Any text of up to 80 characters long.

Line 2: *ifanis, tref, ifdeck*
$ifanis = 1$ for an anisotropic (transversely isotropic) model, $= 0$ for isotropic. $tref$ is the reference period (seconds) of the model for the physical dispersion correction. If $tref \leq 0$, no correction is made. The parameter $ifdeck$ defines the type of model. If $ifdeck = 1$, the model is presented in tabular form. If $ifdeck = 0$, the model is presented as a polynomial.

3.1.2.1 *Tabular Setting,* $ifdeck = 1$.

Line 3: *N, nic, noc*
N is the number of model knots and $N \leq 350$. *nic* is the index of the solid side of the inner core boundary (ICB). *noc* is the index of the fluid side of the mantle core boundary (MCB). **Format:** Unformatted.

Lines 4 − (N + 3): *r, rho, vpv, vsv, qkappa, qshear, vph, vsh, eta*
Each line describes the model parameter set for a single knot at radius r. Note that each knot has an integer index starting from 1, where the index is equal to the line number minus 3. The discontinuity interfaces are defined by a pair of knots at the same radius.
The line fields are:

r - radius of the knot in meters (m). **minos_bran** truncates the fractional part of radius, so a layer with thickness less that 1 m does not make any sense. Introducing thin layers (less than 1 m) leads to creating additional interfaces. If the model includes a surface liquid layer, it must be a single layer without intermediate knots. Radius starts from zero (index= 1) and ends at the free surface, growing from the center of the Earth outward.

rho - density, (kg/m^3)

vpv - velocity of vertically polarized P wave, (m/s)

vsv - velocity of vertically polarized S wave, (m/s)

qkappa - compressional Q

qshear - shear Q

vph - velocity of horizontally polarized P, (m/s)

vsh - velocity of horizontally polarized S, (m/s)

eta - transversely isotropic model parameter

If the model is isotropic, $ifanis = 1$ and **minos_bran** reads and changes the values of *vph, vsh* and *eta* field in the following way: $vph = vpv$, $vsh = vsv$, and $eta = 1$. If both *qkappa* and *qshear* are equal to zero, the Q model is not specified for the knot and no correction for attenuation is made. **Format:** (f8.0, 3f9.2, 2f9.1, 2f9.2, f9.5)

3.1.2.2 *Polynomial Setting,* $ifdeck = 0$.

Line 3: *nreg, nic, noc, rx*
nreg is the number of regions in the model, *nic* and *noc* have the same meaning as for the tabular model, rx is the normalizing radius for the polynomials. rx is given in km and it is usually 6371 km. **Format:** Unformatted.

Line 4: *nlay, r1, r2*
nlay is the number of levels (layers) to be used in the region extending from radius $r1$ to $r2$; $r1$ to $r2$ are in km. **Format:** Unformatted.

Lines 5-9 or 5-12: Lines 4-9 or 4-12 must be repeated *nreg* times. Lines 4-9 must be repeated for the isotropic model and lines 4-12 for the anisotropic model. Each line consists of five coefficients $a_0,\ a_1,\ a_2,\ a_3,\ a_4$ for the fourth order polynomial $P(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + a_4x^4$, where $x = r/rx$. A set of 5 or 8 polynomials is used for interpolating the model parameters: density, velocities, etc., as a function of normalized radius inside the region defined by line 4. **Format:** (5f9.5). More detail:

Line 5: Coefficients for density *rho*, (g/cm^3)

Line 6: Coefficients for *vpv*, (km/s)

Line 7: Coefficients for *vsv*, (km/s)

Line 8: Coefficients for *qkappa*

Line 9: Coefficients for *qshear*

The next three lines must be added for an anisotropic model.

Line 10: Coefficients for *vph*, (km/s)

Line 11: Coefficients for *vsh*, (km/s)

Line 12: Coefficients for *eta*

3.1.3 Output Data

minos_bran outputs an ASCII listing file and a FORTRAN unformatted binary file. The first file contains the model table and the normal mode properties, and the second contains the eigenfunctions.

3.1.3.1 *out_plain_file*

The file consists of two parts:

Part 1: Model table. The model output is always in tabular form. If the input model is in tabular form, the output model is just a copy except that knots are given indices and columns are placed in a different order. For an isotropic model, *vph* will be replaced on *vpv*, *vsh* on *vsv*, and $eta = 1$. If the input model is in polynomial form, the model is converted to tabular form by interpolation across region layers. For each region, the program constructs $nlay + 1$ knots with a constant step in radius. The total number of knots N is equal to $nreg \cdot (nlay + 1)$. Lines from the beginning of the file look as follows:

Lines 1-5: Contains the model *title*, reference period $tref$, header for the model table, and empty lines. **Format:** no format.

Lines $6 - (N+5)$**:** *index, r, rho, vpv, vph, vsv, vsh, eta, dshear, qkappa*
where *index* is the knot number, starting from 1. The content of the remaining fields is described in Section 3.2.2. **Format:** (3x, i3, f12.1, 5f12.2, f12.5, 2f12.2)

Lines $N+6 \div N+11$**:** Contains text messages with Runge-Kutta precision integration, gravity cut off frequency, empty lines, and the header for the normal mode properties table (totaling 6 lines). **Format:** Free.

Part 2: Mode properties. For a fixed radial order number n and angular order l, **minos_bran** computes the eigenfunctions (stored separately) and the scalar parameters (properties): eigenvalue (frequency), phase and group velocities, Q and ratio of kinetic to potential energy. These scalar parameters are stored in the normal mode properties table. Each row (line) in this table gives the properties for the current n and l. Lines following the first part look as follows:

Lines $(N+12)$ **– end:** *norder, typeo, lorder, phvel, freq, per, grvel, Q, raylquo,*
where

norder - mode radial order number n
typeo - type of oscillation (single character): s - spheroidal, t - toroidal, c - inner core toroidal
lorder - mode angular order number l
phvel - phase velocity, (km/s)
freq - frequency (eigenvalue), (millihertz,mHz)
per - period, per = 1000/freq, (sec)
grvel - group velocity, (km/s)
Q - shear Q
raylquo - ratio of kinetic to potential energy minus 1 which should be small (of order *eps*) if the eigenfunction is accurate and if there are enough radial knots in the model to allow quadratures to be done accurately. (You will probably see some degradation in this parameter for strongly exponential modes such as Stoneley modes.)

Format: (i5, a2, i5, 6g16.7)

3.1.3.2 *out_bin_file*

This is a fixed record length binary encapsulated file. The file is not portable, which means that the type of encapsulation strongly depends on the compiler. For example, if the file was created on a SUN platform by the f77 compiler, it cannot be read by a progam created by the g77 compiler on the same platform. To avoid this inconvenience, the program **eigcon** removes encapsulation and creates a binary file portable between different languages and platforms. **minos_bran** outputs each eigenfunction with a single write statement:

```
real*4 abuf(nvec)
.....
write(ioeig) (abuf(i),i=1,nvec)
```

where nvec is $5 + 6 * N$ words long for spheroidal modes and $5 + 2 * N$ words long for other modes. The first five words of `abuf` are n, l, ω, Q, and group velocity C_g. The rest of `abuf` contains eigenfunctions and their derivatives by radius - U(1..N), U'(1..N), V(1..N), V'(1..N), P(1..N), P'(1..N) for spheroidal modes and W(1..N), W'(1..N) for other modes. Here, U is the eigenfunction for the vertical spheroidal (S) component, V for the horizontal S component, P is the gravitational potential. W is the common notation for modes other than S.

Eigenfunctions are stored in normalized form. The normalization of eigenfunctions is given by

$$\omega^2 \int_0^{r_n} \rho(r)\, W^2(r) r^2\, dr = 1$$

for toroidal modes and

$$\omega^2 \int_0^{r_n} \rho(r)\ [U^2(r) + V^2(r)] r^2\ dr = 1$$

for spheroidal modes.

Note that r, ω, ρ, V, U, W in the two formulas above are normalized such that a density of $\rho_n = 5515$ kg/m^3 is 1, πg is 1, where G is the gravitational constant $G = 6.6723 \cdot 10^{-11}\ m^3/kg/s^2$, and the radius r_n of the free surface is 1. These normalizations result in:

acceleration normalization:	$a_n = 10^{20}/(\rho_n r_n^4)$;
velocity normalization:	$v_n = r_n\ (\pi G \rho_n)^{-1/2}$;
frequency normalization:	$\omega_n = v_n / r_n$;
radius normalization:	r_n.

3.1.4 Messages

Program **minos_bran** prints out dialog messages only on the standard output. See an example in Section 8.1.1 (Interactive dialog).

3.2 eigcon Program

The **eigcon** program makes postprocessing of the **minos_bran** results and creates the *.eigen* relation table in *dbname.eigen*, where *dbname* is the database name. The program creates a directory called *dbname.eigen.dat* in the same directory where the file *dbname.eigen* is located. Under *dbname.eigen.dat* it creates a binary file with a fixed name, *eigen*, for storing the eigenfunctions. This binary file consists of segments with a fixed length – one segment per eigenfunction. The *eigen* binary file is a portable file. Unlike the **minos_bran** binary file it does not have any encapsulation and can be easily treated by programs written in various high-level languages such as C, C++, Perl, etc. Access to a segment is provided through the *.eigen* relation table, namely, by referencing the path, length, byte offset, and type of data. The *.eigen* relation table (6.1) has the structure of an external database file and can be easily incorporated into relational databases such as ORACLE, Postgress, MySQL, etc.

3.2.1 Command Line

There are three different ways to start the **eigcon** program:

1. Interactive dialog for setting input parameters:

   ```
   eigcon
   ```

2. Single shell command with redirection of the standard input from a parameter file *parameter_file* containing exactly the same information and in the same order as in the interactive dialog – one answer per line:

   ```
   eigcon < parameter_file
   ```

3. Direct shell script. In this case, the contents of the parameter file are directly included in the shell script between delimiters "WORD":

   ```
   eigcon << WORD
    ........
    ........
   WORD
   ```

See Chapter 8 for examples.

The parameter file consists of six lines:

Line 1: *jcom*
jcom is the type of oscillation. *jcom* = 1 for radial modes, = 2 for toroidal modes, = 3 for spheroidal modes, and = 4 for inner-core toroidal modes. **Format:** Unformatted.

Line 2: *model_file*
model_file is the path to the 1D input model file. **Format:** Text string up to 256 characters long.

Line 3: *max_depth*
max_depth is the depth to cut all output eigenfunctions (in km). All output values exist in the interval $(r_n, r_n - max\_depth)$ only. r_n is the radius of the free surface in km. **Format:** Unformatted.

Line 4: *in_plain_file*
This is the path to the input ASCII file. The file is the output model listing *out_plain_file* of the program **minos_bran**. See Section 3.1.3. **Format:** Text string up to 256 characters long.

Line 5: *in_bin_file*
in_bin_file is the path to the input FORTRAN binary unformatted file, which was produced by the program **minos_bran**. See Section 2.1.

Line 6: *dbname*

dbname is the path to the output database name. The path is a string up to 256 characters long. The path should not end with a backslash, "/". The part of the string after the last "/" (or from the beginning of the string, if the string does not have "/" at all) is the database name. The database name must be at least one character long.

3.2.2 Input Data

model_file See description in Section 3.1.2, ***model_file***.

in_plain_file This file has been created by **minos_bran** as an output file. See description of the file in Section 3.1.3. Actually, **eigcon** uses only normal mode properties from this file to create some part of **.eigen** relation table.

in_bin_file This file has been created by **minos_bran** as an output file. See description of the file in Section 3.2.1.

3.2.3 Output Data

As mentioned above, **eigcon** creates three objects in the file system: a relational table *dbname.eigen*, a directory *dbname.eigen.dat*, and a binary file *dbname.eigen.dat/eigen*.

***dbname.eigen* file format** is described as the relation table *.eigen* (Table 6.1). Each line of this file describes the modal properties of a single eigenfunction and references the binary data segment in the *dbname.eigen.dat/eigen* file.

***dbname.eigen.dat/eigen* binary file** consists of segments. Each segment stores normalized eigenfunctions for a single normal mode (n, l). It consists of words 4 bytes long containing real*4 or integer*4 variables in the internal computer format. The field *datatype* in *.eigen* describes numeric memory storage data (byte order). There are two common byte orders for most computer architectures: BIG_ENDIAN (straight order 1-2-3-4) and LOW_ENDIAN (reverse order 4-3-2-1). BIG_ENDIAN is used by SUN and RISC-oriented platforms, and LOW_ENDIAN is used by PC, VAX and DEC hardware. **eigcon** automaticaly detects the byte order and places the text strings "t4" (BIG_ENDIAN) or "f4" (LOW_ENDIAN) into the *datatype* field in the *.eigen* relation.

Each segment is logically divided into two parts – the header and the body.

3.2.3.1 *The header*

This part of the segment stores scalar parameters and normalization coefficients for the modal properties.

word 1: `integer*4`**,** n Normal mode radial order number n. Must be equal to *norder* in the referencing *.eigen* relation.

word 2: `integer*4`**,** l Normal mode angular order (harmonic degree) number l. Must be equal to *lorder* in the referencing *.eigen* relation.

word 3: `real*4`**,** ω Normal mode frequency (eigenvalue) in physical units rad/s, $\omega = 2\pi/per$. *per* is the period field in the *.eigen* relation.

word 4: `real*4`**,** q Part of the exponential term in the attenuation expression, e^{-qt}, $q = 0.5 * \omega/Q$, rad/sec.

word 5: `real*4`**,** r_n The radius normalization coefficient in meters, equal to the radius of the model's free surface.

word 6: `real*4`**,** v_n The velocity normalization coefficient, $v_n = r_n\ (\pi G \rho_n)^{-1/2}$, where G is the gravitational potential constant ($G = 6.6723 \cdot 10^{-11}$ m^3/kg/s^2), ρ_n is the density normalization coefficient ($\rho_n = 5515$ kg/m^3). The circular frequency normalization coefficient ω_n is given by $\omega_n = v_n/r_n$.

word 7: `real*4`**,** a_n The acceleration normalization coefficient, $a_n = 10^{20}/(\rho_n r_n^4)$.

3.2.3.2 *The body - eigenfunction grid*

word 8 – end: `real*4`, $E(nrow, ncol)$ Matrix of $nrow$ rows and $ncol$ columns to store the normalized eigenfunctions. The first column is radius. The other columns are the eigenfunctions. The number of eigenfunctions depends on modal type. For spheroidal modes, it should be 7 columns (r, U, $U^{'}$, V, $V^{'}$, P, $P^{'}$) for toroidal and others, it should be 3 columns (r, W, $W^{'}$). The matrix is stored in segments by the second index, by columns.

Note that **eigcon** performs an additional eigenfunction normalization. Eigenfunctions for toroidal and the horizontal part of spheroidal modes are divided by $(l(l+1))^{1/2}$. This is done in accordance with theory developed by Woodhouse and Dahlen (1978) [10]. The new normalization of eigenfunctions is given by

$$\omega^2 \int_0^{r_n} \rho(r)\ l(l+1)\ W^2(r) r^2\ dr = 1$$

for toroidal modes and

$$\omega^2 \int_0^{r_n} \rho(r)\ [U^2(r) + l(l+1)\ V^2(r)] r^2\ dr = 1$$

for spheroidal modes.

3.2.4 Messages

Program **eigcon** prints out on the standard output device the following messages:

3.2.4.1 *Dialog messages*

Copy of input/output dialog or parameter file in dialog form. For example,

```
spheroidals (3) or toroidals (2) or radial (1) or
 inner core toroidals (4) modes
3
enter name of model file
model_PREM.txt
enter max depth [km] :
40.
enter name of minos_bran output text file
PREM_S
minos_bran output binary unformatted file name
ePREM_S
enter path/dbase_name or dbase_name to store eigenfunctions:
test_S
```

3.2.4.2 *Info and error messages*

1. `============ Program eigcon ====================`

   Info message. Shows that program **eigcon** starts.

2. `eigcon:  n,nstart,nrad = nnn, mmm, kkk`

   Info message. `nnn` is the total number of nodes of eigenfunction, `mmm` is the number of the starting node after cutting eigenfunction by depth, and `kkk` is the rest number of nodes after cutting.

3. `ERR001:  eigcon:  Input plane and binary files differ:  nn, ll, n, l`

   Order numbers of some eigenfunctions `nn` and `ll` in the text file differ from the numbers `n, l` in the binary data segment. Program is terminated. Check *.eigen* relation or create it again.

4. `ERR002: eigcon: Unknown jcom nnn`

   Error message. The parameter `jcom` with value `nnn` is out of range. Program is terminated. Provide `jcom` with right value.

5. `ERR003: eigcon: jcom = nnn does not fit mode sss`

   Error message. Impossible combination of jcom and mode. For example, jcom=2 (toroidal modes) cannot be used together with a **minos_bran** unformatted file for spheroidal modes. Program is terminated. Provide right jcom and unformatted file.

6. `ERR004:eigcon: Wrong minos_bran output text file`

   Error message. Probably the **minos_bran** plain file was edited. Never change this file. Program is terminated. Rerun **minos_bran** again to create a proper plain file.

3.3 green Program

The **green** program computes the Green's functions for a single event and a given set of stations. For each station, a station-channel list specifying the orientation of the sensors must be input. The program can optionally compute Green's functions (a) for the Z component alone, (b) for 3 components with the standard ZNE sensor orientation (see Appendix C), or (c) for 3 components with the Z component directed "up" and two orthogonal horizontal components oriented arbitrarily. To set up station information, it is necessary to create a flat file database consisting of the two relations *.site*, and *.sitechan* in the CSS 3.0 database format. It is preferable to create the relations by a program, for example, by IRIS's **rdseed** program or by selection from a global database. Other input includes the eigenfunctions which are collected from various databases containing *.eigen* relations for spheroidal or toroidal modes. For each sensor, **green** computes and stores in an output *.wfdisc* relation six Green's functions, one per moment tensor component.

3.3.1 Command Line

There are three different ways to start the **green** program:

1. Interactive dialog for setting the input parameters:

   ```
   green
   ```

2. Single shell command with redirection of the standard input to a parameter file *parameter_file* containing exactly the same information and in the same order as in the interactive dialog – one answer per line

   ```
   green < parameter_file
   ```

3. Direct shell script. In this case, the contents of the parameter file are directly included in the shell script between the delimiters "WORD":

   ```
   green << WORD
    ........
    ........
   WORD
   ```

See Chapter 8 for examples.

The input parameter file consists of six lines:

Line 1: *in_dbname*
in_dbname is the input database name for the *.site* and *.sitechan* relations. **Format:** Any string up to 256 characters long.

Line 2: *db_list*
This is the path to the file defining the list of database names containing the eigenfunctions, one name per line. Each name refers to the database in which the *.eigen* relation resides. **Format:** Any string up to 256 characters long.

Line 3: *cmt_event*
This is the path to the file with the CMT solution for a single event. It includes the CMT location, the seismic moment tensor components, the scalar moment, the focal planes, the source half duration time, and the output time step of the synthetic seismograms. **Format:** Any string up to 256 characters long.

Line 4: *fmin, fmax*
fmin, fmax define the frequency range to be selected from the input eigenfunction databases. All modes with frequencies out of this range are rejected. **Format:** Unformatted.

Line 5: *nsamples*
This is the number of samples in the synthetic seismograms. All synthetic seismograms start from the source time. **Format:** Unformatted.

Line 6: *out_ dbname*
This is the output database name including only the *.wfdisc* relation for unit Green's functions. Each row of the *.wfdisc* relation refers to the binary file with 6 multiplexed Green's functions, i.e., one tensor component. **Format:** Any string up to 256 characters long.

3.3.2 Input Data

in_ dbname This database must exist and must include *.site* and *.sitechan* relations. The *.sitechan* relation provides the station-channel list used by the program. Using these relations, the **green** program defines for each station a certain number of channel groups. Each group consists of the single Z-component channel or a triple of channels (Z-component, and two horizontal components with sensor directions defined by the *hang* field in the *.sitechan* relation). Before grouping, the **green** program sorts the *.sitechan* file by its (*sta, chan*) fields. After sorting, it also removes all duplicate rows (equal station and channel code). Note that the channel field, *chan*, in the *.sitechan* relation is a three-character channel name following the SEED format convention. The first two characters define the type of unique channel (e.g., BH, LH, etc.) for each station's group. The third character must be the component name: Z (uppercase, for vertical) and other symbols (any case, for the horizontals).

db_ list This file allows joining an unlimited number of **eigcon** outputs for spheroidal, toroidal, or radial modes. The single requirement is that intersection between these files is null. Each desired eigenfunction must be present and unique in some **eigcon** output.

cmt_ event This file consists of a single line with the following fields:

evid, year, jday, hour, min, sec, lat, lon, depth, step, halfd,
M_0, M_{rr}, $M_{\theta\theta}$, $M_{\varphi\varphi}$, $M_{r\theta}$, $M_{r\varphi}$, $M_{\theta\varphi}$,
M_n, *strike1, dip1, slip1, strike2, dip2, slip2*

where

evid is the event identifier, 8 characters long

year is the event year in the form **yyyy**

jday is the event day starting from the beginning of the year

hour is the event hour

min is the event minutes

sec is the event seconds with decimal fractions

lat is the event geographical latitude (degree)

lon is the event geographical longitude (degree)

depth is the event depth (km)

step is the time step for Green's functions and seismograms (sec)

halfd is the source half-time duration (sec)

M_0 is the scalar tensor moment (dyn·cm)

M_{rr}, $M_{\theta\theta}$, $M_{\varphi\varphi}$, $M_{r\theta}$, $M_{r\varphi}$, $M_{\theta\varphi}$ are moment tensor components normalized by the coefficient M_n

M_n is the normalization coefficient for the tensor components (dyn·cm)

strike1, dip1, slip1 are the first fault plane solution (degree)

strike2, dip2, slip2 are the second fault plane solution (degree)

3.3.3 Output Data

out_dbname Program **green** creates a *.wfdisc* relation in database *out_dbname* and a multiplexed binary file containing the Green's function waveforms. The binary file contains six Green's functions in an order corresponding to the tensor components: M_{rr}, $M_{\theta\theta}$, $M_{\varphi\varphi}$, $M_{r\theta}$, $M_{r\varphi}$, $M_{\theta\varphi}$. The total length, *nsamp*, of the file is equal to $6 * nsamples$.

3.3.4 Messages

Program **green** prints out on the standard output device the following messages:

3.3.4.1 *Dialog messages*

Copy of input/output dialog or parameter file in dialog form. For example,

```
enter path to db with sta & stachan:
RDSEED_rdseed
enter name of file within list of nmodes db:
db_list
enter input CMT file name:
cmt_event
min and max frequencies to be considered (mHz) :
0. 260.
enter # pts in greens fns .le. 30000 :
4000
enter Green functions output db file name:
green
```

3.3.4.2 *Warning, error, and info messages.*

1. `============= Program green ====================`

   Info message. Shows that program **green** starts.

2. `WARNING: green: # of points in Green functions is stripped to nnn`

   Warning message. The number of requested points for the Green's functions exceeds the maximum allowed value `mseis`. The number is reduced to `mseis` and the program continues to run. To increase the maximum value, change `mseis` to a new bigger value in the `green.h` header and recompile the **green** program. By default, `mseis=30000`.

3. `WARNING: green: # of channels is stripped to 3`

   Warning message. Number of sensors (channels, components) in a group cannot be greater than 3. Only the first three sensors are taken into account.

4. `WARNING: green: # of channels is stripped to 1`

   Warning message. Number of sensors in a group cannot be equal to 2. Only the first sensor is taken into account.

5. `WARNING: green: Channel: # 1 is not vertical. Sequence ignored`

   Warning message. Channel with Z component must be the first in the group. The group of channels is rejected.

6. `WARNING: green: Channel: # nnn is not horizontal. Sequence ignored`

   Warning message. Second and third channels in the group must be horizontal components. nnn is the channel number in the group. The group of channels is rejected.

7. `ERR010: green: max l = nnn, must be .le. ml`

   Error message. The max angular order number `nnn` for some mode `n` exceeds the parameter `ml` set up in the `green.h` header. Program is terminated. To avoid the problem, make `ml` bigger than `nnn` in the `green.h` header and recompile the program. It is not recommended to make `ml` greater than 10000, which leads to inaccuracy in computing the associated Legendre polynomials. By default, `ml = 6000`.

8. `ERR011:eigen: flat and bin indices are different`

   Error message. Broken input *.eigen* relation. Program is terminated. Check or create *.eigen* again.

9. `ERR012: green: # sph. modes in band exceed max allowed number nnn`

   Error message. The total number of spheroidal modes in the band exceeds the maximum allowed number of `nnn`. Program is terminated. Increase parameter `meig` in the `green.h` header and recompile the program. By default, `meig = 200000`.

   `ERR012: green: # tor. modes in band exceed max allowed number nnn`

   Error message. The total number of toroidal modes in the band exceeds the maximum allowed number `nnn`. Program is terminated. Increase parameter `meig` in the `green.h` header and recompile the program. By default, `meig = 200000`.

10. `green: # sph. modes in band = nnn must be .le. meig`

    Info message. `nnn` is the total number of spheroidal modes in all input databases. `meig` is the maximum allowed number of spheroidal modes.

    `green: # tor. modes in band = nnn must be .le. meig`

    Info message. `nnn` is the total number of toroidal modes in all input databases. `meig` is the maximum allowed number of toroidal modes.

11. `green: evid date&time lat = ±dd.ddd, lon = ±ddd.ddd`

    `green: source depth = ddd.d km`

    `green: step = d.ddd sec, nsamples = ddddd`

    Info message. This message outputs part of the *cmt_ event* file: event information, Green's function's time step and number of samples. See description of the *cmt_ event* file in Section 3.3.2.

12. `green: Input dbname : xxxxx`

    Info message. xxxxx is the name of the input database for the *.site* and *.sitechan* relations.

13. `green: Station: code lat lon , Channels: nnn`

    Info message. This message specifies the station code, station coordinates, and number of components nnn for the selected group of sensors.

14. `green: Channel: # nnn code chan hang vang`

    Info message. This message specifies a separate sensor (channel) for the group of sensors. nnn is the current number in the group, code is the station code, chan is the channel name. hang (horizontal angle) and vang (vertical angle) are sensor orientation in space. See *.sitechan* relation, Table 6.4.

15. `green: Epicentral Distance : ddd.ddd`

    Info message. ddd.ddd is the station-event distance in degrees.

16. `green: Azimuth of Source : ddd.ddd`

    Info message. ddd.ddd is the azimuth of a station to the source point in degrees.

17. `green: nnn code chan date&time step mmm`

    Info message. This message informs about some attributes specifying the Green's functions. nnn is the global number of Green's functions set, code is the station name, chan is the channel name, date&time is the event origin time, step is the Green's function sampling step in seconds, and mmm is the total number of samples for the block of six Green's functions.

3.4 syndat Program

The **syndat** program makes synthetic seismograms by convolution of Green's functions (**green** program output) with the seismic moment tensor of the chosen event.

3.4.1 Command Line

There are three different ways to start the **syndat** program:

1. Interactive dialog for setting input parameters:

   ```
   syndat
   ```

2. Single shell command with redirection of the standard input from a parameter file *parameter_file* containing exactly the same information and in the same order as in the interactive dialog - one answer per line:

   ```
   syndat < parameter_file
   ```

3. Direct shell script. In this case, the contents of the parameter file are directly included in the shell script between the delimiters "WORD":

   ```
   syndat << WORD
    ........
    ........
    WORD
   ```

See Chapter 8 for examples.

The parameter file consists of five lines:

Line 1: *cmt_event*
This is the path to the file with the CMT solution for a single event. It must be the same *cmt_event* file as used in the construction of the Green's functions by the **green** program. It uses part of the file defining the seismic moment tensor components, scalar moment, and focal planes. **Format:** Any string up to 256 characters long.

Line 2: *in_dbname*
in_dbname is the input database name for the input Green's functions. The *.wfdisc* relation of the *in_dbname* database must be the output *.wfdisc* relation of the **green** program. **Format:** Any string up to 256 characters long.

Line 3: *plane*
This defines the method for introducing the moment tensor components. If $plane = 0$, the tensor components come in directly from the *cmt_event* file as is. If $plane = 1$, the program takes the scalar moment and angles from focal plane 1 and computes the moment tensor components. For $plane = 2$, the program computes the tensor components using the scalar moment and focal plane 2. **Format:** Unformatted.

Line 4: *out_dbname*
This is the output database name for the *.wfdisc* relation for synthetic seismograms. Each row of the *.wfdisc* relation refers to a binary file with synthetic data in nm/sec. **Format:** Any string up to 256 characters long.

Line 5: *datatype*
This defines the type of synthetic data. If $datatype = 0$ the output synthetic waveforms are accelerograms in nm/s^2. This is recommended native **Mineos** output. If $datatype = 1$ the output is velocity waveform in nm/s, and if $datatype = 2$ the output is displacement in nm. This is done by additional conversion of accelerograms to velocity or displacement.

3.4.2 Input Data

cmt_ event See description in Section 3.3.2.

in_ dbname *out_ dbname* database for the *.wfdisc* relation from the **green** program.

3.4.3 Output Data

out_ dbname Database name for the final synthetic seismograms referenced by the *.wfdisc* relation.

Chapter 4

The Utilities User's Manual

4.1 cucss2sac

NAME

cucss2sac Converts CSS 3.0 waveforms to SAC binary or ASCII files

SYNOPSIS

cucss2sac [-a [-n]] db_name out_SAC_dir

OPTIONS

-a Generate ASCII files

-n Suppress header output in ASCII files. Used together with **-a** option.

DESCRIPTION

The **cucss2sac** utility converts a CSS 3.0 *.wfdisc* relation into a set of SAC binary files or ASCII files. Output files are stored in **out_SAC_dir** directory. This program has been designed especially for the **Mineos** package due to bugs in the standard IRIS **css2sac** utility. **cucss2sac** supports limited capabilities and is not a complete substitution of the standard **css2sac** utility. For example, **cucss2sac** only supports the CSS3.0 schema and two input CSS binary data formats:

- "t4" (BIG_ENDIAN, single precision floating point)
- "f4" (LOW_ENDIAN, single precision floating point)

cucss2sac converts CSS binary data "as is" without any corrections or modifications. For the **Mineos** package, **cucss2sac** may be used for the conversion of Green's functions or synthetic seismograms to binary SAC or ASCII files.

A CSS database **db_name** must include at least one file `db_name.wfdisc` file containing CSS format information about the waveforms. It is also desirable to add to the database *.origin* and *.site* relation tables (files `db_name.origin` and `db_name.site`) to get a more complete SAC header. Note that the *.origin* relation keeps information about event location and the *.site* relation keeps station locations. If *.origin* is present in the database, **cucss2sac** uses only the first line from this file. In the case of ASCII output, all available parameters go to the ASCII header – three lines in the beginning of the file. For example, an output file for a synthetic seismogram looks as follows:

```
EVENT: 2000(014)-23:37:10.800 947893030.80000 25.3900 101.4000 33.0000
STATION: ALE LHE 82.5033 -62.3500 0.0000
DATA: 2000(014)-23:37:10.800 947893030.80000 8000 1.000000
```

```
0.0000000E+00 5.7025738E+00
1.0000000E+00 1.9508668E+00
2.0000000E+00 -4.1160989E+00
3.0000000E+00 -4.1094885E+00
......
......
......
7.9970000E+03 1.5211651E+01
7.9980000E+03 2.7408613E+01
7.9990000E+03 3.5795624E+01
```

where

- the `EVENT` line represents event information: origin time (human and epoch), latitude (deg), longitude (deg), and depth (km);
- the `STATION` line represents station information: station code, channel name, latitude (deg), longitude (deg), and depth = 0;
- the `DATA` line represents data parameters: waveform starting time (human and epoch), number of samples, and sampling step (sec).

The rest of the lines (the body) is a two-column table. The first column represents relative time from the beginning of waveform in seconds, and the second one shows waveform samples in nm, nm/s or nm/s^2, according to the **syndat** output format.

If the presence of the header is not desired, suppress it with option **-n** (no header).

The output ASCII file format for the Green's functions is different. The header is the same, but the body is a 7-column table. The first column is relative time, as before, and the other 6 columns are the 6-unit Green's functions ($G_{rr}, G_{\theta\theta}, G_{\varphi\varphi}, G_{r\theta}, G_{r\varphi}, G_{\theta\varphi}$) for the chosen component – vertical or some horizontal. The order of the components is the same as the order of the seismic moment tensor components M_{ij} defined in the **syndat** program. So the corresponding synthetic seismogram S (**NOTE:** IN ACCELERATION, nm/s^2) is given by the formula

$$S = 10^{-18} \sum_{ij} G_{ij}\ M_{ij}$$

where M_{ij} are given in dyne*cm for $ij = rr, \theta\theta, \varphi\varphi, r\theta, r\varphi, \theta\varphi$.

NOTE: According to SAC compliance, event depth is stored in the SAC header in meters.

EXAMPLES

```
cucss2sac Syndat Syndat_SAC
cucss2sac green green_SAC
cucss2sac -a Syndat Syndat_ASC
cucss2sac -a -n Syndat Syndat_ASC_NOHEADER
```

4.2 eigen2asc

NAME

eigen2asc Converts the *.eigen* relation table to ASCII files

SYNOPSIS

eigen2asc [-n] nmin nmax lmin lmax db_name out_dir

OPTIONS

-n Suppress header output in ASCII files.

DESCRIPTION

The **eigen2asc** utility converts some part or the whole *.eigen* relation into a set of ASCII files. Program **eigen2asc** searches in the `db_name.eigen` file for all eigenfunctions with mode numbers n, l satisfying the following conditions: **nmin** $\leq$ n $\leq$ **nmax**, **lmin** $\leq$ l $\leq$ **lmax**, and converts eigenfunctions to ASCII files. Output files are stored in **out_dir**. Each output file consists of the header and the body. The header is a single line including the first 10 fields of the *.eigen* relation, see Table 6.1. The option **-n** excludes the header output. The body is a three-column (radial, toroidal modes) or seven-column (spheroidal mode) table. The first body column is radius in meters and the other columns contain the eigen functions and their derivatives by radius (in the internal, normalized units) – U, U', V, V', P, P' for spheroidal modes and W, W' for other modes. For more details, see Section 4.1. The output file name has the form **X.nnnnnnn.mmmmmmm.ASC**, where, letter **X** is "S" for spheroidal modes or "T" for toroidal modes, **nnnnnnn** is the number n, and **mmmmmmm** is the mode number l.

EXAMPLES

```
eigen2asc 0 1 2 500 test_S Eigen_S_ASC
eigen2asc -n 0 1 2 500 test_S Eigen_S_no_headers_ASC
```

4.3 endi

NAME

endi In-place file swapping with fixed width

SYNOPSIS

endi nw file1 [file2 ... filen]

DESCRIPTION

This utility is used when you transfer binary files of *.wfdisc* or *.eigen* relations to a platform with a different byte order. The utility changes the byte order from BIG_ENDIAN to LOW_ENDIAN and vice versa. **endi** sequentially reads the file from the list **file1 [file2 ... filen]** into memory as an unsigned character string, swaps sequential groups of **nw** bytes long starting from the beginning of the file, and stores the swapped data into the same place as before (swapping in-place). If the last group has a length less than **nw**, it stays unswapped.

EXAMPLES

In this example, w.00001 and w.00002 are the binary real*4 data files from some *.wfdisc* relation created on a PC computer. These files have internal LOW_ENDIAN data representation created, e.g., by a direct access FORTRAN WRITE statement. After applying the command:

```
... > endi 4 w.00001 w.00002
```

we change the order of 4-byte words to BIG_ENDIAN for transferring these files on a computer with BIG_ENDIAN byte order, e.g., a SUN Ultra machine, where we may use a direct access READ statement to read data as real*4 variables.

4.4 simpledit

NAME

simpledit Filter to create *.site* and *.sitechan* CSS relations from an input ASCII file

SYNOPSIS

simpledit ascii_file db_name

DESCRIPTION

The **simpledit** is a simple filter program. It converts a manually created (text editor) unformatted ASCII file **ascii_file** with station and channel information into CSS3.0 **db_name.site** and **db_name.sitechan** relation tables (files). The prefix **db_name**, coming from the command line, is the database name. In the case of real data, there is another way to create these relations. Use the IRIS **rdseed** program, which provides SEED volume conversion to CSS format. Note, CSS format is strictly defined, so do not create *.site* and *.sitechan* tables manually. This leads to numerous format errors and as a result can crash the **Mineos** programs. In case of a lot of station data, a better solution is to write a program which creates **ascii_file** or *.site* and *.sitechan* files directly.

Description of ascii_file The **ascii_file** is a set of ASCII text lines. Each line describes a single station or a single channel. All channel lines, placed after some station line, belong to that station.

For example, consider the following file:

```
ANMO 34.9502 -106.4602 1.6890 'Albuquerque, New Mexico, USA'
@ LH1 150.0000 280.0 90.0
@ LH2 150.0000 10.0 90.0
@ LHZ 89.3000 0.0 0.0
BJT 40.0183 116.1679 0.1370 'Baijiatuan, Beijing, China'
@ LHE 60.0000 90.0 90.0
@ LHN 60.0000 0.0 90.0
@ LHZ 60.0000 0.0 0.0
@ BHE 60.0000 90.0 90.0
@ BHN 60.0000 0.0 90.0
@ BHZ 60.0000 0.0 180.0
BILL 68.0651 166.4524 0.2990 'Bilibino, Russia'
@ LHZ 0.0000 0.0 0.0
CCM 38.0557 -91.2446 0.1710 'Cathedral Cave, Missouri, USA'
@ LHE 51.0000 90.0 90.0
@ LHN 51.0000 0.0 90.0
@ LHZ 51.0000 0.0 0.0
```

Each line in the file consists of some number of fields separated with one or more space characters. If a field has an internal space character, e.g., as in a station's full name, it must be surrounded with single quotes. The first field starts with the first position in the line. A station line has the fields: *sta, lat, lon, elev, staname*. See the description in the *.site* relation table, Table 6.3. A channel line starts with text field "@". The other fields are: *chan, edepth, hang, vang*. See description in Table 6.4, *.sitechan* relation. The standard orientation of a sensor "XX" is defined as

```
@ XXE dd.dddd 90.0 90.0
@ XXN dd.dddd 0.0 90.0
@ XXZ dd.dddd 0.0 0.0
```

where `dd.dddd` is a sensor *edepth*. Note that the order of channels for the given station is not important.

4.5 create_origin

NAME

create_origin Convert an event text file into the CSS3.0 *.origin* relation.

SYNOPSIS

create_origin cmt_event db_name

DESCRIPTION

The shell script **create_origin** reads from the **cmt_event** ASCII file the first line written in the format of input **cmt_event** file for program **green**, converts the source time and location into a single row CSS3.0 relation table, and stores it into the **db_name.origin** file. The script **create_origin** only fills out in the *.origin* relation the following fields: *lat, lon, depth, time, orid, jdate, auth*, and *lddate*. The other fields are not important for the **Mineos** package and are filled with default values. For more details about the *.origin* relation see Table 4.1 below, or for the complete description, see Anderson et al. 1990 [1].

Table 4.1: Relation: **origin**. Description: Data on event location and confidence bounds.

attribute name	field no.	storage type	external format	character position	attribute description
lat	1	f4	f9.4	1-9	estimated latitude
lon	2	f4	f9.4	11-19	estimated longitude
depth	3	f4	f9.4	21-29	estimated depth
time	4	f8	f17.5	31-47	epoch time
orid	5	i4	i8	49-56	origin id
evid	6	i4	i8	58-65	event id
jdate	7	i4	i8	67-74	julian date
nass	8	i4	i4	76-79	no. of associated phases
ndef	9	i4	i4	81-84	no. of locating phases
ndp	10	i4	i4	86-89	no. of depth phases
grn	11	i4	i8	91-98	geographic region no.
srn	12	i4	i8	100-107	seismic region no.
etype	13	c7	a7	109-115	event type
depdp	14	f4	f9.4	117-125	est. depth from depth phases
dtype	15	c1	a1	127-127	depth method used
mb	16	f4	f7.2	129-135	body wave magnitude
mbid	17	i4	i8	137-144	mb magid
ms	18	f4	f7.2	146-152	surface wave magnitude
msid	19	i4	i8	154-161	ms magid latitude
ml	20	f4	f7.2	163-169	local magnitude
mlid	21	i4	i8	171-178	ml magid
algorithm	22	c15	a15	180-194	location algorithm used
auth	23	c15	a15	196-210	source/originator
commid	24	i4	i8	212-219	comment id
lddate	25	date	a17	221-237	load date

Chapter 5

`fdb` and `time` Subroutines/Functions

5.1 `fdb` Subprograms

The **fdb** (fortran database) subroutines/functions come from the CU-Boulder **fdb** FORTRAN 77 library. They provide a FORTRAN interface to *.site, .sitechan, .wfdisc*, and *.eigen* relations. Most of **fdb** subroutines/functions have very short and simple source code. The list of function names with a brief description is listed below:

subroutine read_site reads *.site* relation from file

subroutine write_site writes *.site* relation to file

subroutine default_site sets up default (empty) tuple of *.site* relation in memory

subroutine read_sitechan reads *.sitechan* relation from file

subroutine write_sitechan writes *.sitechan* relation to file

subroutine default_sitechan sets up default (empty) tuple of *.sitechan* relation in memory

subroutine read_wfdisc reads *.wfdisc* relation from file

subroutine write_wfdisc writes *.wfdisc* relation to file

subroutine put_wfdisc writes for selected *.wfdisc* tuple associated binary file

subroutine get_wfdisc reads for selected *.wfdisc* tuple associated binary file

subroutine select_sitechan grouping *.sitechan* relation by components (single Z component or 3-components)

subroutine default_wfdisc sets up default (empty) tuple of *.wfdisc* relation in memory triple of channels)

subroutine open_eigen opens an *.eigen* file and associated binary file

subroutine close_eigen closes an *.eigen* file and associated binary file

subroutine write_eigen writes a single tuple into the *.eigen* file

subroutine read_eigen reads a single tuple from the *.eigen* file

subroutine null_eigen sets up default (empty) tuple of *.eigen* relation in memory

subroutine get_eigen reads current *.eigen* relation's tuple associated binary file

subroutine put_eigen writes current *.eigen* relation's tuple associated binary file

integer*4 factor2 factors an integer number into two factors

5.2 time Subprograms

The **time** subroutines/functions perform conversions from human to epoch time and vice versa. Function **loctime** returns local system time as a text string. A short description follows:

subroutine epochtoh (t, year, doy, hour, min, sec) converts epoch time, **double* t**, into human time **year,** day of year (**doy**)**, hour**, minutes (**min**), **sec**. Note: **integer*4** applies to **year, doy, hour, min** and **real*8** applies to **sec.** Day of year **(doy)** is the number of days counted from the first day of the year.

real*8 function htoepoch (year, doy, hour, min, sec) returns epoch time. Input arguments are **year, doy, hour**, minutes (**min**), **sec**. Note: **integer*4** applies to **year, doy, hour, min** and **real*8** applies to **sec.**

subroutine doytom (year, doy, mon, day) converts **year** and day of year (**doy**) into month (**mon**) and day of month (**day**). The type of all arguments is **integer*4**.

integer*4 function mtodoy (year, mon, day) returns days of year. Input arguments are **year**, month (**mon**), and **day**. The type of all arguments is **integer*4**.

character*17 function loctime() returns local system time of form **mm/dd/yy-hh:mm:ss**.

Chapter 6

Flat File Database Tables

This chapter defines the tables (relations) in the flat file database. The name of each relation appears in bold print at the top of each table. The format of the flat files specify fixed field widths and precisions in FORTRAN style. Fields are separated in these files with exactly one blank space.

Table 6.1: Relation: **eigen**. Description: Eigenfunction and eigenvalue file header.

attribute name	field no.	storage type	external format	character position	attribute description
norder	1	i4	i8	1-8	radial order no. n
lorder	2	i4	i8	10-17	angular order no. l
typeo	3	i4	a1	19-19	type of modes
eigid	4	i4	i8	21-28	eigen id
per	5	f4	f16.5	30-45	eigenvalue, period
phvel	6	f4	f16.5	47-62	phase velocity
grvel	7	f4	f16.5	64-79	group velocity
attn	8	f4	f16.5	81-96	attenuation
nrow	9	i4	i8	98-105	no. of rows
ncol	10	i4	i4	107-110	no. of columns
npar	11	i4	i4	112-115	no. of parameters
datatype	12	c2	a2	117-118	numeric storage
foff	13	i4	i10	120-129	byte offset
dir	14	c64	a64	131-194	directory
dfile	15	c32	a32	196-227	file name
commid	16	i4	i8	229-236	comment id
lddate	17	date	a17	238-254	load date

Table 6.3: Relation: **site**. Description: Station location information.

attribute name	field no.	storage type	external format	character position	attribute description
sta	1	c6	a6	1-6	station identifier
ondate	2	i4	i8	8-15	Julian start date
offdate	3	i4	i8	17-24	Julian off date
lat	4	f4	f9.4	26-34	latitude
lon	5	f4	f9.4	36-44	longitude
elev	6	f4	f9.4	46-54	elevation
staname	7	c50	a50	56-105	station description
statype	8	c4	a4	107-110	station type: single station, array, etc.
refsta	9	c6	a6	112-117	reference station for array members
dnorth	10	f4	f9.4	119-127	offset from array reference (km)
deast	11	f4	f9.4	129-137	offset from array reference (km)
lddate	12	date	a17	139-155	load date

Table 6.4: Relation: **sitechan**. Description: Station-channel information.

attribute name	field no.	storage type	external format	character position	attribute description
sta	1	c6	a6	1-6	station identifier
chan	2	c8	a8	8-15	channel identifier
ondate	3	i4	i8	17-24	Julian start date
chanid	4	i4	i8	26-33	channel id
offdate	5	i4	i8	35-42	Julian off date
ctype	6	c4	a4	44-47	channel type
cdepth	7	f4	f9.4	49-57	emplacement depth
hang	8	f4	f6.1	59-64	horizontal angle
vang	9	f4	f6.1	66-71	vertical angle
descrip	10	c50	a50	73-122	channel description
lddate	11	date	a17	124-140	load date

Table 6.5: Relation: **wfdisc**. Description: Waveform file header and descriptive information.

attribute name	field no.	storage type	external format	character position	attribute description
sta	1	c6	a6	1-6	station identifier
chan	2	c8	a8	8-15	channel identifier
time	3	f8	f17.5	17-33	epoch time of the first sample in file
wfid	4	i4	i8	35-42	waveform id
chanid	5	i4	i8	44-51	channel operation id
jdate	6	i4	i8	53-60	Julian date
endtime	7	f8	if17.5	62-78	time*(nsamp-1)/samprate
nsamp	8	i4	i8	80-87	no. of samples
samprate	9	f4	f11.5	89-99	sampling rate in samples/sec
calib	1	f4	f16.6	101-116	nominal calibration
calper	11	f4	f16.6	118-133	nominal calibration period
instype	12	c6	a6	135-140	instrument code
segtype	13	c1	a1	142-142	indexing method
datatype	14	c2	a2	144-145	numerical storage
clip	15	c1	a1	147-147	clipped flag
dir	16	c64	ia64	149-212	directory
dfile	17	c32	a32	214-245	data file
foff	18	i4	i10	247-256	byte offset
commid	19	i4	i8	258-265	comment id
lddate	20	date	a17	267-283	load date

Chapter 7

Attribute Description

Name: *attn*

Relation: eigen

Description: Attenuation coefficient Q

NA Value: -1

Range: $attn > 0$

Name: *calib*

Relation: wfdisc

Description: Calibration factor. This is the conversion factor that maps digital data to earth displacement. The factor holds true at the oscillation period specified by the attribute *calper*. A positive value means ground motion increasing in a component direction (up, north, east) is indicated by increasing counts. A negative value means the opposite. *Calib* generally reflects the best calibration information available at a time of recording.

NA Value: NOT ALLOWED. A valid entry is required.

Units: Nanometers/digital count

Range: Any nonzero floating point number

Name: *clip*

Relation: wfdisc

Description: Clipped data flag. This is a single-character flag to indicate whether (c) or not (n) the data were clipped.

NA Value: - (a dash)

Range: { c | n}, lowercase

Name: *calper*

Relation: wfdisc

Description: Calibration period. This gives the period for which *calib* is valid.

NA Value: NOT ALLOWED. A valid entry is required.

Units: Seconds

Range: $calper > 0$

Name: *chan*

Relation: sitechan, wfdisc

Description: Channel identifier. This is a three-character code, which taken together with *sta, jdate* and *time*, uniquely identifies the source of seismic data, including the geographic location, spatial orientation, sensor and subsequent data processing. The first two character define the channel name, and the third, the channel component. "Z" character is for the vertical component; other characters are for horizontal components.

NA Value: A valid entry is required.

Range: Any sequence of 3 uppercase characters.

Name: *chanid*

Relation: wfdisc

Description: Channel recording identifier. This is the surrogate key used to uniquely identify a specific recording. *Chanid* duplicates the information of the compound key *sta, chan, time*. As a single identifier it is often convenient. *Sta, chan, time* is more appropriate to the human interface.

NA Value: -1

Range: $chanid > 0$

Name: *commid*

Relation: eigen, wfdisc

Description: Data file. Comment identification. This is the integer key used to point to free-form comments in the predefined remark list. Not implemented yet.

NA Value: -1 NO ANY REMARKS.

Range: $commid \geq 0$

Name: *ctype*

Relation: sitechan

Description: This attribute specifies the type of data channel: n (normal , a normal instrument response), b (beam, a coherent beam firmed with array data), or i (an incoherent beam of energy stack).

NA Value: - (a dash)

Range: { n | b | i }, lowercase

Name: *datatype*

Relation: eigen, wfdisc

Description: Numeric storage data. This attribute specifies the format of data series in the file system. Currently only the data type t4 and f4 are supported. Attribute is used to enable swapping four bytes in real numbers if necessary.

NA Value: NOT ALLOWED. A valid entry is required.

Range: The currently recognized types (lowercase) are:

legal datatype values		
datatype value	size (byte)	description
t4	4	SUN IEEE single precision real
f4	4	PC/DEC/VAX IEEE single precision real

Name: *descrip*

Relation: sitechan

Description: Channel description. This is a description of the data channel. For non-instrument channels (e.g., beams) this is the only quantitative description of channel operations in the core tables.

NA Value: - (a dash)

Range: Any free-format string up to 50 characters long.

Name: *deast*

Relation: site

Description: Distance east. This attribute gives the "easting" or relative position of an array element, east of the array center specified by the value of *refsta*. See *dnorth*.

NA Value: 0.0

Units: Kilometers

Range: $-20,000.00 \leq deast \leq +20,000.00$

Name: *dfile*

Relation: eigen, wfdisc

Description: Data file. In $wfdisc$ this is the file name of a disk waveform or single component Green's function. In *eigen* this is the name of a multiplexed eigenfunction file.

NA Value: NOT ALLOWED. A valid entry is required.

Range: Any string up to 32 characters long.

Name: *dir*

Relation: eigen, wfdisc

Description: Directory. This attribute is the directory part of a path name. This is a relative path for the current directory containing the *eigen* relation.

NA Value: NOT ALLOWED. A valid entry is required.

Range: Any string of up to 64 characters.

Name: *dnorth*

Relation: site

Description: Distance north. This attribute gives the "northing" or relative position of an array element, north of the array center specified by the value of *refsta*. See *deast*.

NA Value: 0.0

Units: Kilometers

Range: $-20,000.00 \leq dnorth \leq +20,000.00$

Name: *edepth*

Relation: sitechan

Description: Emplacement depth. This attribute gives the depth at which the instrument is positioned, relative to the value of *elev* in the **site** relation.

NA Value: NOT ALLOWED. A valid entry is required.

Units: Kilometers

Range: $edepth \geq 0.0$

Name: *eigid*

Relation: eigen

Description: Eigenfunction identifier. The key field is a unique identifier for an eigenfunction data defined by mode numbers n, l and the type of mode.

NA Value: NOT ALLOWED. A valid entry is required.

Name: *elev*

Relation: site

Description: Elevation. This attribute is the elevation of a seismic station relative to mean sea level.

NA Value: -999.0

Units: Kilometers

Range: $-10.0 \leq elev \leq +10.0$

Name: *endtime*

Relation: wfdisc

Description: Time of last datum. This attribute is the time of the last sample in the waveform file. *Endtime* is equivalent to $time + (nsamp - 1)/samprate$.

NA Value: +9999999999.999

Range: $endtime > time$

Name: *foff*

Relation: eigen, wfdisc

Description: File offset. This is the byte offset of eigenfunction segments within data file. See *dir* and *dfile*.

NA Value: NOT ALLOWED. A valid entry is required.

Range: $foff \geq 0$

Name: *grvel*

Relation: eigen

Description: Group velocity. This gives the magnitude value of group velocity for attribute *per*. Evaluated only for S, T, and C modes. See *typeo*.

NA Value: -1

Units: kilometers/seconds

Range: $grvel > 0$

Name: *hang*

Relation: sitechan

Description: Horizontal orientation of seismometer. This attribute specifies the orientation of the seismometer in the horizontal plane, measured clockwise from North. For a North-South orientation with the seismometer pointing toward the north, $hang = 0$; for East-West orientation with seismometer pointing toward the west, $hang = 270$. See *vang*.

NA Value: NOT ALLOWED. A valid entry is required.

Units: Degrees

Range: $0.0 \leq vang \leq 360.0$

Name: *instype*

Relation: wfdisc

Description: Instrument type. This character string is used to indicate the instrument type. Some examples are: SRO, ASRO, and S-750.

NA Value: - (a dash)

Range: Uppercase, and too numerous to mention. For details, see Ganse and Hutt, 1982 [4].

Name: *jdate*

Relation: wfdisc

Description: Julian date. This attribute is the date of seismic recording. The same information is available in epoch time, but the Julian date format is more convenient for many types of searches. Dates B.C.E. are negative. Note: there is no year = 0000 or day = 000. Where only the year is known, day of the year = 001; where only year and month are known, day of year = first day of month. For example, Jan 1 of 10 B.C.E. is -0010001. See *time* and attribute *per*. Evaluated only for S, T, and C modes. See *typeo*.

NA Value: -1

Range: Julian dates of the form yyyyddd. Must be consistent with the accompanying *time* attribute.

Name: *lat*

Relation: site

Description: Latitude. This attribute is the geographic latitude. Locations north of the equator have positive latitudes.

NA Value: NOT ALLOWED. A valid entry is required.

Units: Degrees

Range: $-90.0 \leq lat \leq +90.0$

Name: *lddate*

Relation: all

Description: Load date. This is the date and time the record was created. Not implemented yet.

Range: Any value

Name: *lon*

Relation: site

Description: Longitude. This attribute is the geographic longitude. Longitudes are measured positive east of the Greenwich meridian.

NA Value: NOT ALLOWED. A valid entry is required.

Units: Degrees

Range: $-180.0 \leq lon \leq +180.0$

Name: *lorder*

Relation: eigen

Description: Angular order number (harmonic degree) l of a normal mode.

NA Value: -1

Name: *ncol*

Relation: eigen

Description: This attribute is the number of columns in the eigenfunction grid stored in binary data.

FORTRAN: i4

NA Value: NOT ALLOWED. A valid entry is required.

Range: { 2 | 4 | 6 }

Name: *norder*

Relation: eigen

Description: Radial order number n of a normal mode.

NA Value: -1

Name: *npar*

Relation: eigen

Description: This attribute is the number of parameters stored in the binary data.

NA Value: 0

Range: $npar > 0$

Name: *nrow*

Relation: eigen

Description: This attribute is the number of rows in the eigenfunction grid stored in the binary data.

NA Value: NOT ALLOWED. A valid entry is required.

Range: $nrow > 0$

Name: *nsamp*

Relation: wfdisc

Description: Number of samples. This quantity is the number of samples in waveform segment.

NA Value: NOT ALLOWED. A valid entry is required.

Range: $nsamp > 0$

Name: *offdate*

Relation: site, sitechan

Description: Turn off date. This attribute is the Julian Date on which the station or sensor indicated was turned off, dismantled, or removed. *See ondate.*

NA Value: -1

Range: Julian date of the form yyyyddd

Name: *ondate*

Relation: site, sitechan

Description: Turn on date. This attribute is the Julian Date on which the station or sensor indicated began operation. *Offdate* or *ondate* is not intended to accommodate temporary downtimes, but rather to indicate the time period for which the attributes of the station *lat, lon, elev* are valid for the given station code. Stations are often moved, but usually the station code remains unchanged.

NA Value: NOT ALLOWED. A valid entry is required.

Range: Julian date of the form yyyyddd

Name: *per*

Relation: eigen

Description: Eigenvalue period. The frequency is $\omega = 2 * \pi/per$. The normalized value of ω is located in the associated binary data.

NA Value -1

Units: seconds

Range: $per > 0$

Name: *phvel*

Relation: eigen

Description: Phase velocity. This gives the magnitude value of phase velocity for attribute *per*. Evaluated only for S, T, and C modes. See *typeo*.

NA Value: -1

Units: kilometers/seconds

Range: $phvel > 0$

Name: *samprate*

Relation: wfdisc

Description: Sampling rate. The attribute is the sample rate in samples/second. This value may vary slightly from the nominal to reflect clock drift.

NA Value: NOT ALLOWED. A valid entry is required.

Units: 1/seconds

Range: $samprate > 0$

Name: *segtype*

Relation: wfdisc

Description: Segment type. The attribute indicates if a waveform is o (original), v (virtual), s (segmented), d (duplicate), g (Green's function), or w (synthetic waveform).

NA Value: - (a dash)

Range: { o | v | s | d | g | w }

Name: *sta*

Relation: site, sitechan, wfdisc

Description: Station code, or common code-name for a seismic observatory. Generally only three or four characters are used.

NA Value: A valid entry is required.

Range: Any uppercase string of up to 6 characters.

Name: *staname*

Relation: site

Description: Station name/description. This is the full name of the station whose code-name is in *sta*.

NA Value: - (a dash)

Range: Any uppercase string of up to 50 characters.

Name: *statype*

Relation: site

Description: Station type. This character string specifies the station type. Recommended types are ss (single stations) or ar (array).

NA Value: - (a dash)

Range: { ss | ar }, lowercase

Name: *time*

Relation: wfdisc

Description: Epoch time. Epochal time given as seconds and fractions of a second since hour 0 January 1, 1970, and stored in a double precision floating number. Refers to start time data. The double precision floating number allows 15 decimal digits. At 1 millisecond accuracy this is a range of 3×10^4 years. Generally only three or four characters are used.

NA Value: NOT ALLOWED. A valid entry is required.

Units: Seconds

Name: *typeo*

Relation: eigen

Description: Type of binary data. This single character indicates the type of data. The values are R (radial), S (spheroidal), T (toroidal) or C (inner core toroidal) modes. The character P indicates that data includes constants, normalization parameters, and radius samples.

NA Value: NOT ALLOWED. A valid entry is required.

Name: *refsta*

Relation: site

Description: Reference station. This string specifies the reference station with respect to which the array members are located. See *deast, dnorth*.

NA Value: - (a dash)

Range: Any *sta* from **site**.

Name: *vang*

Relation: sitechan

Description: Vertical orientation of seismometer. This attribute measures the angle between the sensitive axis of a seismometer and the outward-pointing vertical direction. For a vertically oriented seismometer, $vang = 0$. For a horizontally oriented seismometer, $vang = 90$. See *hang*.

NA Value: NOT ALLOWED. A valid entry is required.

Units: Degrees

Range: $0.0 \leq vang \leq 90.0$

Chapter 8

Examples

This section contains examples of the interaction with each of the four programs of the **Mineos** package. Each program can be run in three different ways: by interactive dialog, redirection from an input file, or direct use of a shell script. In each subsection, examples of running the program in each way are presented. The output from each approach should be the same. The programs are run in order: **minos_bran**, **eigcon**, **green**, **syndat**.

All of the input files named here are included in the standard distribution of the **Mineos** package.

8.1 minos_bran

In the example given here, **minos_bran** reads in a model file called *prem_noocean.txt*, which is a tabular listing of the PREM model with the ocean filled in with solid crust. The program outputs two files: *prem_noocean_S* and *eprem_noocean_S*. The first file contains a listing of some normal mode properties (n, l, frequency, period, phase and group speed, etc.) and the second file contains the eigenfunctions. Both files are needed to be read into the eigenfunction renormalization program, **eigcon**. In this example, the numerical tolerance parameter eps is set to 10^{-10} and gravity is taken into consideration in computing the eigenfunctions only if the frequency is less than 10 mHz. The output normal mode properties will be for spheroidal modes with angular order l values ranging between 1 and 6000, frequencies ranging between 0 and 166 mHz (i.e., 6 sec), and radial orders n ranging from 0 to 0 – that is, only fundamental modes will be computed in this example.

8.1.1 *Example 1.* Interactive dialog

```
$ minos_bran
input model file:
prem_noocean.txt
output file:
prem_noocean_S
eigenfunction file (output):
eprem_noocean_S
enter eps and wgrav
1e-10 10
enter jcom (1=rad;2=tor;3=sph;4=ictor)
3
enter lmin,lmax,wmin,wmax,nmin,nmax
1 6000 0.0 166.0 0 0
$
```

8.1.2 *Example 2.* Redirection of input file

Create in the working directory a parameter file named `Param` with the following contents:

```
prem_noocean.txt
prem_noocean_S
eprem_noocean_S
1e-10 10
3
1 6000 0.0 166.0 0 0
```

and start the following command:

```
$ minos_bran < Param
```

8.1.3 *Example 3.* Direct shell script

Include in your sh/csh script the following lines:

```
.............
minos_bran << EOF
prem_noocean.txt
prem_noocean_S
eprem_noocean_S
1e-10 10
3
1 6000 0.0 166.0 0 0
EOF
.............
```

8.2 eigcon

In this example, the two files computed by **minos_bran** are read in as input, *prem_noocean_S* and *eprem_noocean_S*, together with the input model file *prem_noocean.txt*. **eigcon** renormalizes the eigenfunctions and outputs them to a depth of 1000 km, in this example. The renormalized eigenfunctions are placed in an extension of the CSS3.0 database, using the relation *test_S.eigen*. This relation points to a file called *eigen* located in a subdirectory called *test_S.dat*. The file *eigen* is non-encapsulated, which allows greater flexibility in access from different platforms and code from different compilers. Information about the eigenfunction's byte order is contained in the *.eigen* relation, which is used in subsequent programs to swap bytes appropriately. Eigenfunctions are computed here to 1000 km for plotting purposes, but for runs in which earthquakes are no deeper than, for example, 40 km, "40" would be input here.

8.2.1 *Example 1.* Interactive dialog

```
$ eigcon
spheroidals (3) or toroidals (2) or radial (1) or
inner core toroidals (4) modes
3
enter name of model file
prem_noocean.txt
enter max depth [km] :
1000
enter name of minos_bran output text file
prem_noocean_S
minos_bran output binary unformatted file name
```

```
eprem_noocean_S
enter pathdbase_name or dbase_name to store eigenfunctions:
test_S
$
```

8.2.2 *Example 2.* Redirection of input file

Create in the working directory a parameter file named `Param` with the following contents:

```
3
prem_noocean.txt
1000
prem_noocean_S
eprem_noocean_S
test_S
```

and start the following command:

```
$ eigcon < Param
```

8.2.3 *Example 3.* Direct shell script

Include in your sh/csh script the following lines:

```
.............
eigcon << EOF
3
prem_noocean.txt
1000
prem_noocean_S
eprem_noocean_S
test_S
EOF
.............
```

8.3 green

In this example, there are three principal input files.

1. The first input file is the database name of the *.site* and *.sitechan* relations. The entries of the *.sitechan* file determine which stations and channels are used for constructing the Green's functions. Channel orientations are in the *.sitechan* file, but station coordinates are in the *.site* file. The database name for these relations in this example is *short*. There must be, therefore, two pre-constructed files: *short.site* and *short.sitechan*.

2. The second input file is the file *db_ list*. This file contains the listing of all database names for the eigenfunction files. In the previous subsection, the *.eigen* relation *prem_ noocean_ S.eigen* was created. So if that file contains the only normal modes to be used in the construction of the Green's functions, then the file *db_ list* would have a single entry: prem_noocean_S.

   > **Note:** Only the database name is included and not the relation name suffix *.eigen*. If other modes are desired, then the file *db_ list* would include the database names of the other modes. For example, toroidal modes are usually included in synthetic, or SH motions will be ignored. In this simple example, *db_ list* can be considered to have a single entry.

3. The third input file is the file *china_ cmt_ evt*, which is a single-lined listing containing the coordinates and event parameters of an earthquake in China. The moment tensor is not used by this program, but by the program **syndat** which follows.

This example will choose modes only between frequencies of 0 and 166 mHz (i.e., periods greater than 6 sec). It will produce Green's functions that are 8000 samples long. The time sampling specified in the *china_ cmt_ evt* file is 1 sec, so this is a time series length of a little over two hours. In many cases, both the minor and major arc arrivals can be seen.

The program will output a *.wfdisc* relation in the database called *green*; that is, a file called *green.wfdisc* which points to the waveforms on disk in a default location.

8.3.1 *Example 1.* Interactive dialog

```
$ green
enter path to db with sta & stachan:
short
enter name of file within list of nmodes db:
db_list
enter input CMT file name:
china_cmt_event
min and max frequencies to be considered (mHz) :
0 166.
enter # pts in greens fns .le. 30000 :
8000
enter Green functions output db file name:
green
$
```

8.3.2 *Example 2.* Redirection of input file

Create in the working directory a parameter file with the name `Param` with the following contents:

```
short
db_list
china_cmt_event
0. 166.
8000
green
```

and start the following command:

```
$ green < Param
```

8.3.3 *Example 3.* Direct shell script

Include in your sh/csh script the following lines:

```
.............
green << EOF
short
db_list
china_cmt_event
0. 166.
8000
green
EOF
.............
```

8.4 syndat

In this example, **syndat** reads in two files. First, there is the event file that contains the CMT: *china_ cmt_ event*. Second, there is the output from the program **green**: *green.wfdisc*. Only the database name, *green*, is input rather than the whole file name. The output database name is also specified, which in this example is *Syndat*. The program will output a *.wfdisc* relation, *Syndat.wfdisc* in this example, pointing to the waveforms on disk.

8.4.1 *Example 1.* Interactive dialog

```
$ syndat
enter input CMT file name:
china_cmt_event
enter tensor type: 0 - moment, 1 - nodal plane 1, 2 - nodal plane 2
0
enter input dbname
green
enter output dbname
Syndat
enter output datatype: 0 -accn, 1 -vel, 2 -displ
0
$
```

8.4.2 *Example 2.* Redirection of input file

Create in working directory parameter file with the name `Param` with the following contents:

```
china_cmt_event
0
green
Syndat
0
```

and start the following command:

```
$ syndat < Param
```

8.4.3 *Example 3.* Direct shell script

Include in your sh/csh script the following lines:

```
.............
syndat << EOF
china_cmt_event
0
green
Syndat
0
EOF
.............
```

Bibliography

[1] Anderson, J., Farrell, W.E., et al. Center for Seismic Studies version 3 database: Schema reference manual. Technical Report C90-01, DARPA, September 1990.

[2] Biswas, N.N., and L. Knopoff (1970), Exact earth-flattening calculations for Love wave. *Bull. Seismol. Soc. Amer.*, *60*, 1123-1127.

[3] Biswas, N.N. (1972), Earth-flattening procedure for propagation of Rayleigh wave. *PAGEOPH*, *96*, 61-74.

[4] Ganse, R., and C.R. Hutt (1982), *Directory of World Digital Seismic Stations.* Boulder: World Data Center for Solid Earth Geophysics, Report SE-32.

[5] Gilbert, F., and A.M. Dziewonski (1975), An application of normal mode theory to the retrieval of structural parameters and source mechanisms from seismic spectra. *Phil. Trans. R. Soc.*, *A278*, 187-269.

[6] Gilbert, F., and G.E. Backus (1969), A computational problem encountered in a study of the earth's normal modes. *Proceedings of AFIPS Fall Joint Computer Conference, 32,* San Francisco, CA, December 1968, 1273-1277.

[7] Gilbert, F., and G.E. Backus (1966), Propagator matrices in elastic wave and vibration problems. *Geophys. 31(2),* 326-332.

[8] Herrmann, R.B. (1978), *Computer programs in earthquake seismology.* Vol. 2, St. Louis: St. Louis University.

[9] Shanks, E.B. (1966), Solutions of differential equations by evaluations of functions, *Math. Comp. 20*, 21-38 MR 32:4858

[10] Woodhouse, J.H., and F.A. Dahlen (1978), The effect of a general aspherical perturbation on the free oscillation of the Earth. *Geophys. J.R. Astr Soc.*, *53*, 335-354.

[11] SPECFEM3D_GLOBE, Version 3.6. User Manual. CIG/CIT, October 24, 2006

Appendix A

Model Example

One-dimensional aniziotropic **PREM** noocean model. The **PREM** water layer is filled with solid crust. The columns in the model table have the following names:

radius	rho	vpv	vsv	qkappa	qshear	vph	vsh	eta
PREM MODEL: (0.0000 0.000) anisotropic case								
1 1.00000 1								
185 33 66								
0.	13088.50	11262.20	3667.80	1327.7	84.6	11262.20	3667.80	1.00000
38172.	13088.18	11261.97	3667.64	1327.7	84.6	11261.97	3667.64	1.00000
76344.	13087.23	11261.29	3667.16	1327.7	84.6	11261.29	3667.16	1.00000
114516.	13085.64	11260.14	3666.36	1327.7	84.6	11260.14	3666.36	1.00000
152688.	13083.42	11258.54	3665.25	1327.7	84.6	11258.54	3665.25	1.00000
190859.	13080.57	11256.49	3663.81	1327.7	84.6	11256.49	3663.81	1.00000
229031.	13077.08	11253.98	3662.05	1327.7	84.6	11253.98	3662.05	1.00000
267203.	13072.95	11251.01	3659.98	1327.7	84.6	11251.01	3659.98	1.00000
305375.	13068.19	11247.58	3657.58	1327.7	84.6	11247.58	3657.58	1.00000
343547.	13062.80	11243.70	3654.87	1327.7	84.6	11243.70	3654.87	1.00000
381719.	13056.77	11239.35	3651.83	1327.7	84.6	11239.35	3651.83	1.00000
419891.	13050.11	11234.56	3648.48	1327.7	84.6	11234.56	3648.48	1.00000
458062.	13042.81	11229.30	3644.81	1327.7	84.6	11229.30	3644.81	1.00000
496234.	13034.88	11223.59	3640.82	1327.7	84.6	11223.59	3640.82	1.00000
534406.	13026.31	11217.42	3636.51	1327.7	84.6	11217.42	3636.51	1.00000
572578.	13017.11	11210.80	3631.88	1327.7	84.6	11210.80	3631.88	1.00000
610750.	13007.28	11203.72	3626.93	1327.7	84.6	11203.72	3626.93	1.00000
648922.	12996.81	11196.18	3621.66	1327.7	84.6	11196.18	3621.66	1.00000
687094.	12985.70	11188.18	3616.07	1327.7	84.6	11188.18	3616.07	1.00000
725266.	12973.97	11179.73	3610.16	1327.7	84.6	11179.73	3610.16	1.00000
763438.	12961.59	11170.82	3603.94	1327.7	84.6	11170.82	3603.94	1.00000
801609.	12948.58	11161.45	3597.39	1327.7	84.6	11161.45	3597.39	1.00000
839781.	12934.94	11151.63	3590.53	1327.7	84.6	11151.63	3590.53	1.00000
877953.	12920.66	11141.35	3583.34	1327.7	84.6	11141.35	3583.34	1.00000
916125.	12905.75	11130.61	3575.84	1327.7	84.6	11130.61	3575.84	1.00000
954297.	12890.21	11119.42	3568.01	1327.7	84.6	11119.42	3568.01	1.00000
992469.	12874.02	11107.76	3559.87	1327.7	84.6	11107.76	3559.87	1.00000
1030641.	12857.21	11095.66	3551.41	1327.7	84.6	11095.66	3551.41	1.00000
1068812.	12839.76	11083.09	3542.63	1327.7	84.6	11083.09	3542.63	1.00000

1106984.	12821.67	11070.07	3533.53	1327.7	84.6	11070.07	3533.53	1.00000
1145156.	12802.96	11056.59	3524.11	1327.7	84.6	11056.59	3524.11	1.00000
1183328.	12783.60	11042.65	3514.37	1327.7	84.6	11042.65	3514.37	1.00000
1221500.	12763.61	11028.26	3504.31	1327.7	84.6	11028.26	3504.31	1.00000
1221500.	12166.33	10355.72	0.00	57823.0	0.0	10355.72	0.00	1.00000
1292078.	12129.26	10314.43	0.00	57823.0	0.0	10314.43	0.00	1.00000
1362656.	12090.47	10272.30	0.00	57823.0	0.0	10272.30	0.00	1.00000
1433234.	12049.91	10229.21	0.00	57823.0	0.0	10229.21	0.00	1.00000
1503812.	12007.54	10185.05	0.00	57823.0	0.0	10185.05	0.00	1.00000
1574391.	11963.32	10139.71	0.00	57823.0	0.0	10139.71	0.00	1.00000
1644969.	11917.20	10093.08	0.00	57823.0	0.0	10093.08	0.00	1.00000
1715547.	11869.14	10045.05	0.00	57823.0	0.0	10045.05	0.00	1.00000
1786125.	11819.08	9995.51	0.00	57823.0	0.0	9995.51	0.00	1.00000
1856703.	11766.99	9944.34	0.00	57823.0	0.0	9944.34	0.00	1.00000
1927281.	11712.82	9891.43	0.00	57823.0	0.0	9891.43	0.00	1.00000
1997859.	11656.52	9836.69	0.00	57823.0	0.0	9836.69	0.00	1.00000
2068438.	11598.05	9779.98	0.00	57823.0	0.0	9779.98	0.00	1.00000
2139016.	11537.37	9721.22	0.00	57823.0	0.0	9721.22	0.00	1.00000
2209594.	11474.42	9660.27	0.00	57823.0	0.0	9660.27	0.00	1.00000
2280172.	11409.17	9597.04	0.00	57823.0	0.0	9597.04	0.00	1.00000
2350750.	11341.57	9531.41	0.00	57823.0	0.0	9531.41	0.00	1.00000
2421328.	11271.58	9463.27	0.00	57823.0	0.0	9463.27	0.00	1.00000
2491906.	11199.14	9392.50	0.00	57823.0	0.0	9392.50	0.00	1.00000
2562484.	11124.21	9319.01	0.00	57823.0	0.0	9319.01	0.00	1.00000
2633062.	11046.76	9242.68	0.00	57823.0	0.0	9242.68	0.00	1.00000
2703641.	10966.73	9163.40	0.00	57823.0	0.0	9163.40	0.00	1.00000
2774219.	10884.07	9081.05	0.00	57823.0	0.0	9081.05	0.00	1.00000
2844797.	10798.75	8995.53	0.00	57823.0	0.0	8995.53	0.00	1.00000
2915375.	10710.72	8906.73	0.00	57823.0	0.0	8906.73	0.00	1.00000
2985953.	10619.93	8814.53	0.00	57823.0	0.0	8814.53	0.00	1.00000
3056531.	10526.34	8718.83	0.00	57823.0	0.0	8718.83	0.00	1.00000
3127109.	10429.91	8619.51	0.00	57823.0	0.0	8619.51	0.00	1.00000
3197688.	10330.58	8516.46	0.00	57823.0	0.0	8516.46	0.00	1.00000
3268266.	10228.31	8409.58	0.00	57823.0	0.0	8409.58	0.00	1.00000
3338844.	10123.06	8298.74	0.00	57823.0	0.0	8298.74	0.00	1.00000
3409422.	10014.79	8183.85	0.00	57823.0	0.0	8183.85	0.00	1.00000
3480000.	9903.44	8064.79	0.00	57823.0	0.0	8064.79	0.00	1.00000
3480000.	5566.46	13716.62	7264.65	57823.0	312.0	13716.62	7264.65	1.00000

3517500.	5547.67	13707.45	7265.03	57823.0	312.0	13707.45	7265.03	1.00000
3555000.	5528.91	13698.36	7265.37	57823.0	312.0	13698.36	7265.37	1.00000
3592500.	5510.18	13689.36	7265.69	57823.0	312.0	13689.36	7265.69	1.00000
3630000.	5491.48	13680.44	7265.97	57823.0	312.0	13680.44	7265.97	1.00000
3630000.	5491.48	13680.42	7265.93	57823.0	312.0	13680.42	7265.93	1.00000
3666482.	5473.29	13636.25	7249.23	57823.0	312.0	13636.25	7249.23	1.00000
3702963.	5455.13	13592.44	7232.63	57823.0	312.0	13592.44	7232.63	1.00000
3739444.	5436.97	13548.96	7216.12	57823.0	312.0	13548.96	7216.12	1.00000
3775926.	5418.82	13505.78	7199.68	57823.0	312.0	13505.78	7199.68	1.00000
3812407.	5400.67	13462.87	7183.30	57823.0	312.0	13462.87	7183.30	1.00000
3848889.	5382.52	13420.19	7166.98	57823.0	312.0	13420.19	7166.98	1.00000
3885370.	5364.37	13377.73	7150.70	57823.0	312.0	13377.73	7150.70	1.00000
3921852.	5346.21	13335.45	7134.44	57823.0	312.0	13335.45	7134.44	1.00000
3958333.	5328.04	13293.32	7118.21	57823.0	312.0	13293.32	7118.21	1.00000
3994815.	5309.85	13251.30	7102.00	57823.0	312.0	13251.30	7102.00	1.00000
4031296.	5291.65	13209.38	7085.78	57823.0	312.0	13209.38	7085.78	1.00000
4067778.	5273.43	13167.51	7069.55	57823.0	312.0	13167.51	7069.55	1.00000
4104259.	5255.18	13125.68	7053.30	57823.0	312.0	13125.68	7053.30	1.00000
4140741.	5236.91	13083.84	7037.02	57823.0	312.0	13083.84	7037.02	1.00000
4177222.	5218.60	13041.98	7020.70	57823.0	312.0	13041.98	7020.70	1.00000
4213704.	5200.26	13000.05	7004.32	57823.0	312.0	13000.05	7004.32	1.00000
4250185.	5181.88	12958.04	6987.89	57823.0	312.0	12958.04	6987.89	1.00000
4286667.	5163.46	12915.90	6971.38	57823.0	312.0	12915.90	6971.38	1.00000
4323148.	5145.00	12873.61	6954.79	57823.0	312.0	12873.61	6954.79	1.00000
4359630.	5126.48	12831.15	6938.10	57823.0	312.0	12831.15	6938.10	1.00000
4396111.	5107.92	12788.47	6921.31	57823.0	312.0	12788.47	6921.31	1.00000
4432593.	5089.29	12745.55	6904.41	57823.0	312.0	12745.55	6904.41	1.00000
4469074.	5070.61	12702.36	6887.38	57823.0	312.0	12702.36	6887.38	1.00000
4505556.	5051.87	12658.87	6870.21	57823.0	312.0	12658.87	6870.21	1.00000
4542037.	5033.06	12615.05	6852.90	57823.0	312.0	12615.05	6852.90	1.00000
4578518.	5014.18	12570.87	6835.43	57823.0	312.0	12570.87	6835.43	1.00000
4615000.	4995.22	12526.30	6817.80	57823.0	312.0	12526.30	6817.80	1.00000
4651482.	4976.19	12481.31	6799.99	57823.0	312.0	12481.31	6799.99	1.00000
4687963.	4957.08	12435.87	6781.99	57823.0	312.0	12435.87	6781.99	1.00000
4724444.	4937.89	12389.94	6763.79	57823.0	312.0	12389.94	6763.79	1.00000
4760926.	4918.61	12343.51	6745.38	57823.0	312.0	12343.51	6745.38	1.00000
4797407.	4899.24	12296.54	6726.76	57823.0	312.0	12296.54	6726.76	1.00000
4833889.	4879.78	12248.99	6707.90	57823.0	312.0	12248.99	6707.90	1.00000
4870370.	4860.21	12200.85	6688.80	57823.0	312.0	12200.85	6688.80	1.00000

4906852.	4840.55	12152.07	6669.46	57823.0	312.0	12152.07	6669.46	1.00000
4943333.	4820.79	12102.63	6649.85	57823.0	312.0	12102.63	6649.85	1.00000
4979815.	4800.91	12052.51	6629.97	57823.0	312.0	12052.51	6629.97	1.00000
5016296.	4780.93	12001.66	6609.81	57823.0	312.0	12001.66	6609.81	1.00000
5052778.	4760.83	11950.06	6589.36	57823.0	312.0	11950.06	6589.36	1.00000
5089259.	4740.61	11897.68	6568.61	57823.0	312.0	11897.68	6568.61	1.00000
5125741.	4720.27	11844.49	6547.54	57823.0	312.0	11844.49	6547.54	1.00000
5162222.	4699.80	11790.46	6526.15	57823.0	312.0	11790.46	6526.15	1.00000
5198704.	4679.21	11735.56	6504.42	57823.0	312.0	11735.56	6504.42	1.00000
5235185.	4658.48	11679.76	6482.35	57823.0	312.0	11679.76	6482.35	1.00000
5271667.	4637.62	11623.02	6459.92	57823.0	312.0	11623.02	6459.92	1.00000
5308148.	4616.62	11565.33	6437.13	57823.0	312.0	11565.33	6437.13	1.00000
5344630.	4595.48	11506.65	6413.97	57823.0	312.0	11506.65	6413.97	1.00000
5381111.	4574.19	11446.94	6390.41	57823.0	312.0	11446.94	6390.41	1.00000
5417593.	4552.75	11386.18	6366.46	57823.0	312.0	11386.18	6366.46	1.00000
5454074.	4531.16	11324.35	6342.10	57823.0	312.0	11324.35	6342.10	1.00000
5490556.	4509.42	11261.40	6317.33	57823.0	312.0	11261.40	6317.33	1.00000
5527037.	4487.51	11197.31	6292.13	57823.0	312.0	11197.31	6292.13	1.00000
5563518.	4465.44	11132.05	6266.48	57823.0	312.0	11132.05	6266.48	1.00000
5600000.	4443.20	11065.59	6240.39	57823.0	312.0	11065.59	6240.39	1.00000
5600000.	4443.20	11065.60	6240.54	57823.0	312.0	11065.60	6240.54	1.00000
5633667.	4422.53	10960.91	6142.04	57823.0	312.0	10960.91	6142.04	1.00000
5667333.	4401.71	10856.15	6043.57	57823.0	312.0	10856.15	6043.57	1.00000
5701000.	4380.74	10751.32	5945.13	57823.0	312.0	10751.32	5945.13	1.00000
5701000.	3992.12	10266.17	5570.21	57823.0	143.0	10266.17	5570.21	1.00000
5771000.	3975.82	10157.76	5516.02	57823.0	143.0	10157.76	5516.02	1.00000
5771000.	3975.82	10157.83	5515.93	57823.0	143.0	10157.43	5515.93	1.00000
5804333.	3933.81	9987.17	5418.69	57823.0	143.0	9986.77	5418.69	1.00000
5837667.	3891.80	9816.52	5321.45	57823.0	143.0	9816.12	5321.45	1.00000
5871000.	3849.78	9645.87	5224.21	57823.0	143.0	9645.47	5224.21	1.00000
5904333.	3807.77	9475.22	5126.97	57823.0	143.0	9474.82	5126.97	1.00000
5937667.	3765.76	9304.57	5029.73	57823.0	143.0	9304.17	5029.73	1.00000
5971000.	3723.75	9133.92	4932.49	57823.0	143.0	9133.52	4932.49	1.00000
5971000.	3543.26	8905.24	4769.90	57823.0	143.0	8905.24	4769.90	1.00000
5993500.	3529.83	8861.96	4754.15	57823.0	143.0	8861.96	4754.15	1.00000
6016000.	3516.39	8818.67	4738.40	57823.0	143.0	8818.67	4738.40	1.00000
6038500.	3502.96	8775.38	4722.65	57823.0	143.0	8775.38	4722.65	1.00000
6061000.	3489.52	8732.10	4706.90	57823.0	143.0	8732.10	4706.90	1.00000
6083500.	3476.08	8688.81	4691.15	57823.0	143.0	8688.81	4691.15	1.00000

6106000.	3462.65	8645.52	4675.40	57823.0	143.0	8645.52	4675.40	1.00000
6128500.	3449.21	8602.24	4659.65	57823.0	143.0	8602.24	4659.65	1.00000
6151000.	3435.77	8558.95	4643.90	57823.0	143.0	8558.95	4643.90	1.00000
6151000.	3359.49	7800.45	4441.09	57823.0	80.0	8048.56	4436.26	0.97646
6179000.	3362.53	7832.17	4434.63	57823.0	80.0	8068.85	4461.39	0.96557
6207000.	3365.58	7863.90	4428.18	57823.0	80.0	8089.15	4486.52	0.95468
6235000.	3368.62	7895.62	4421.73	57823.0	80.0	8109.44	4511.65	0.94379
6263000.	3371.66	7927.34	4415.28	57823.0	80.0	8129.73	4536.78	0.93290
6291000.	3374.71	7959.06	4408.83	57823.0	80.0	8150.02	4561.90	0.92201
6291000.	3374.71	7959.06	4408.83	57823.0	600.0	8150.02	4561.99	0.92201
6318800.	3377.73	7990.56	4402.43	57823.0	600.0	8170.17	4586.94	0.91120
6346600.	3380.75	8022.06	4396.02	57823.0	600.0	8190.32	4611.89	0.90039
6346600.	2900.00	6800.00	3900.00	57823.0	600.0	6800.00	3900.00	1.00000
6347000.	2900.00	6800.00	3900.00	57823.0	600.0	6800.00	3900.00	1.00000
6348000.	2900.00	6800.00	3900.00	57823.0	600.0	6800.00	3900.00	1.00000
6349000.	2900.00	6800.00	3900.00	57823.0	600.0	6800.00	3900.00	1.00000
6350000.	2900.00	6800.00	3900.00	57823.0	600.0	6800.00	3900.00	1.00000
6351000.	2900.00	6800.00	3900.00	57823.0	600.0	6800.00	3900.00	1.00000
6352000.	2900.00	6800.00	3900.00	57823.0	600.0	6800.00	3900.00	1.00000
6353000.	2900.00	6800.00	3900.00	57823.0	600.0	6800.00	3900.00	1.00000
6354000.	2900.00	6800.00	3900.00	57823.0	600.0	6800.00	3900.00	1.00000
6355000.	2900.00	6800.00	3900.00	57823.0	600.0	6800.00	3900.00	1.00000
6356000.	2900.00	6800.00	3900.00	57823.0	600.0	6800.00	3900.00	1.00000
6356000.	2600.00	5800.00	3200.00	57823.0	600.0	5800.00	3200.00	1.00000
6357000.	2600.00	5800.00	3200.00	57823.0	600.0	5800.00	3200.00	1.00000
6358000.	2600.00	5800.00	3200.00	57823.0	600.0	5800.00	3200.00	1.00000
6359000.	2600.00	5800.00	3200.00	57823.0	600.0	5800.00	3200.00	1.00000
6360000.	2600.00	5800.00	3200.00	57823.0	600.0	5800.00	3200.00	1.00000
6361000.	2600.00	5800.00	3200.00	57823.0	600.0	5800.00	3200.00	1.00000
6362000.	2600.00	5800.00	3200.00	57823.0	600.0	5800.00	3200.00	1.00000
6363000.	2600.00	5800.00	3200.00	57823.0	600.0	5800.00	3200.00	1.00000
6364000.	2600.00	5800.00	3200.00	57823.0	600.0	5800.00	3200.00	1.00000
6365000.	2600.00	5800.00	3200.00	57823.0	600.0	5800.00	3200.00	1.00000
6366000.	2600.00	5800.00	3200.00	57823.0	600.0	5800.00	3200.00	1.00000
6367000.	2600.00	5800.00	3200.00	57823.0	600.0	5800.00	3200.00	1.00000
6368000.	2600.00	5800.00	3200.00	57823.0	600.0	5800.00	3200.00	1.00000
6368000.	2600.00	5800.00	3200.00	57823.0	600.0	5800.00	3200.00	1.00000
6369000.	2600.00	5800.00	3200.00	57823.0	600.0	5800.00	3200.00	1.00000
6370000.	2600.00	5800.00	3200.00	57823.0	600.0	5800.00	3200.00	1.00000
6371000.	2600.00	5800.00	3200.00	57823.0	600.0	5800.00	3200.00	1.00000

Appendix B

Benchmarking

Due to modification of the **Mineos** codes, the revised version has been benchmarked again. Three benchmark tests were performed: a test against the original code at UCSD, a test against Bob Hermann's eigenfunction and synthetic seismogram code for fundamental modes, and a test against SPECFEM3D_GLOBE v3.6 code designed for simulation of three-dimensional global seismic wave propagation based upon the spectral-element method (SEM).

B.1 Revised Version vs. UCSD Version

This test consists of computation by both codes of eigenvalues and eigenfunctions for all types of oscillation (spheroidal, toroidal, inner core toroidal, and radial), and, finally, synthetic seismograms for spheroidal, toroidal and radial oscillations. Eigenvalues and eigenfunctions were computed for the first four branches ($n = 0, 1, 2, 3$) in the frequency range from zero to 0.25 Hz. The test showed a perfect matching of the eigenvalues, eigenfunctions and synthetic seismograms for both codes. Differences did not exceed the last significant digit. The most important difference of the revised version against the UCSD code is that the revised code uses sensor orientation "Up" instead of "Down." The revised code uses the coordinate system "Up," "South," and "East," so we need to reverse the sign of all Green's functions and synthetic seismograms of the UCSD code to get the revised code results. Also note that the revised **Mineos** version uses geographic coordinates for the input data; i.e, station and event locations, instead of geocentric ones as in the old version. **Mineos** automatically converts geographic coordinates to geocentric for internal computations.

B.2 Mineos Code vs. Herrmann's Plane Code for Fundamental Modes

The **Mineos** synthetic seismograms were benchmarked against Herrmann's plane code seismograms for the fundamental spheroidal and toroidal modes in the period range 6 - 100 seconds. For testing purposes, 8 1D models were taken: 6 vertical profiles of the global 3D CUB2.0 CU Earth model, PREM model with 3 km water layer, and PREM model with water layer filled with upper crust. The chosen 6 points are located at places characterized by different tectonics, namely:

- Korean Peninsula (36N, 128E)
- Utah, U.S., seismo-tectonic region (40N, 112W)
- Near Hudson bay, Canada, craton (56N, 90W)
- Center of Hudson Bay, Canada, craton (58N, 86N)
- Young Pacific Ocean (0N, 100W)
- Old Pacific Ocean (40N, 160E)

To take into account the Earth's sphericity, the original version of Herrmann's plane code [8] had been modified using the Earth flattening exact formulas for Love waves [2] and Earth flattening approximation for Rayleigh waves [3]. All input information in Herrmann's code is in geocentric coordinates.

As an example, Figures B.1 - B.3 illustrate plane (blue) and spherical (red) three-component synthetic seismograms for the three different seismic stations: BJT, TLY, and BILL. The event location (CMT solution) is 25.39N, 101.40E, depth is 33 km. The station coordinates (geographic), epicentral distances (geocentric), and source azimuths (geocentric) are shown in Table B.1; units are degrees.

Code	Station name	Latitude	Longitude	Distance	Azimuth
BJT	Beijing, China	40.0183N	116.1679E	19.123	-135.267
TLY	Talaya, Russia	51.6807N	103.6438E	26.308	-175.417
BILL	Bilibino, Russia	68.0651N	166.4524E	57.417	-103.266

Table B.1: Station coordinates (geographic), epicentral distances (geocentric), and source azimuths (geocentric) for Benchmark test #2, Mineos vs. Herrmann's plane code for fundamental modes.

Moment tensor components are:

$$M_{rr} = -0.60e24,\ M_{\theta\theta} = -6.29e24,\ M_{\varphi\varphi} = 6.89e24, M_{r\theta} = -1.85e24,\ M_{r\varphi} = 0.12e24,\ M_{\theta\varphi} = -4.73e24$$

The input model is isotropic double-layered crust PREM in which the water layer is filled with the upper crust's velocities.
Computations for both codes were performed without attenuation and gravity effects. Srtictly speaking, plane code does not support gravity computation at all, so gravity was turned off for the **Mineos** code. The **Mineos** synthetic accelerograms were converted to displacement. All seismograms were computed in the period range 5 to 200 seconds. The spectral range was tapered with half-cosine windows with corner frequencies (1/200, 1/100) Hz and (1/6, 1/5) Hz.

The dispersion curves of the phase and group velocities obtained from the two codes (Figure B.4) are practically identical; the maximum absolute difference of velocities doesn't exceed 0.8 m/s. Synthetic seismograms are very close, except the long period in the earlier part of the records. This difference is due to the significant increasing noise level after acceleration-displacement transformation (proportional to $1/\omega^2$) and due to differences in the deeper parts of the input models.

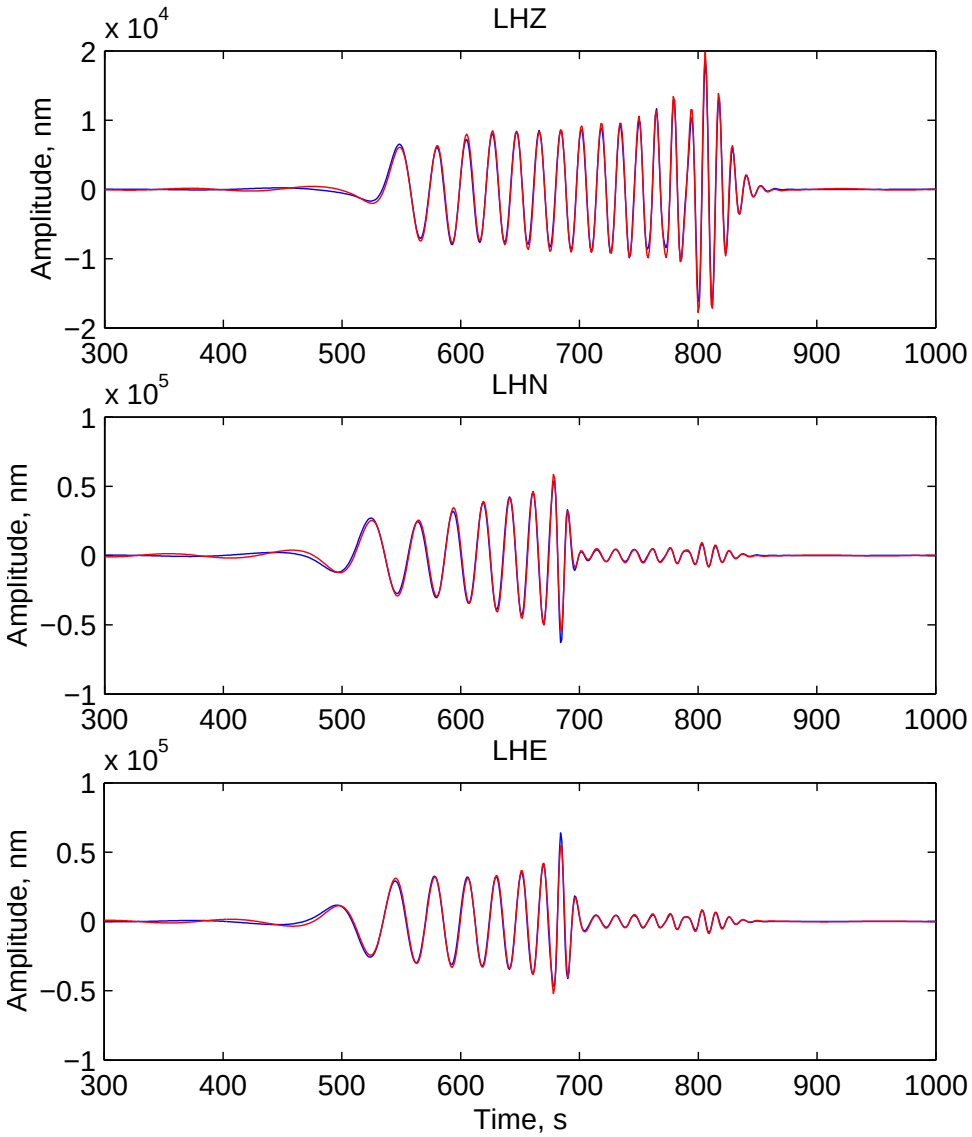


Figure B.1: Station BJT. Comparison with Herrmann's plane code. Three-component synthetic seismogram for the fundamental spheroidal and toroidal modes. **Mineos** seismogram is plotted in red, Herrmann's in blue. Earthquake is 25.39N, 101.40E (Southern China), depth is 33 km. Model is PREM, in which the water layer is filled with the upper crust's velocities. The crust has only two layers.

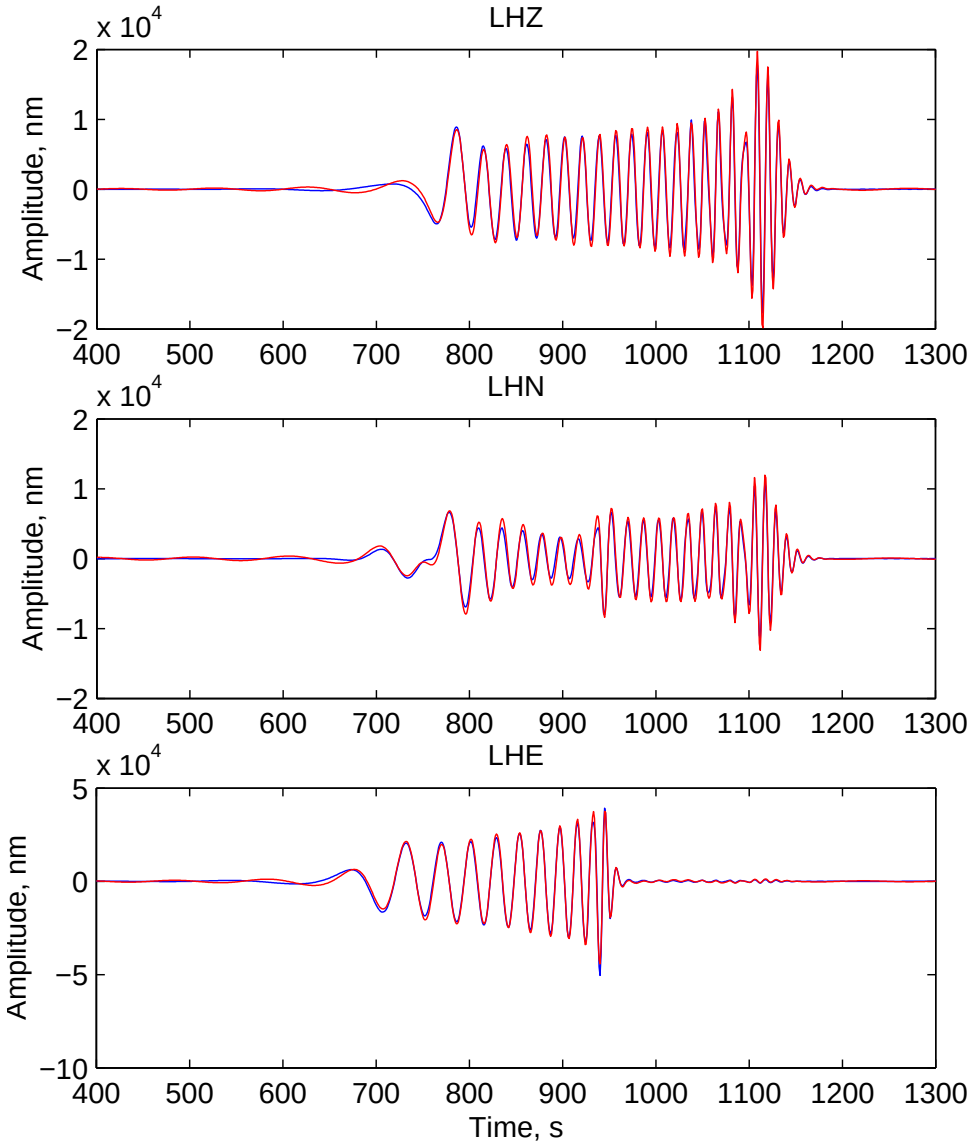


Figure B.2: Comparison with Herrmann's plane code, as in Figure B.1, but for station TLY.

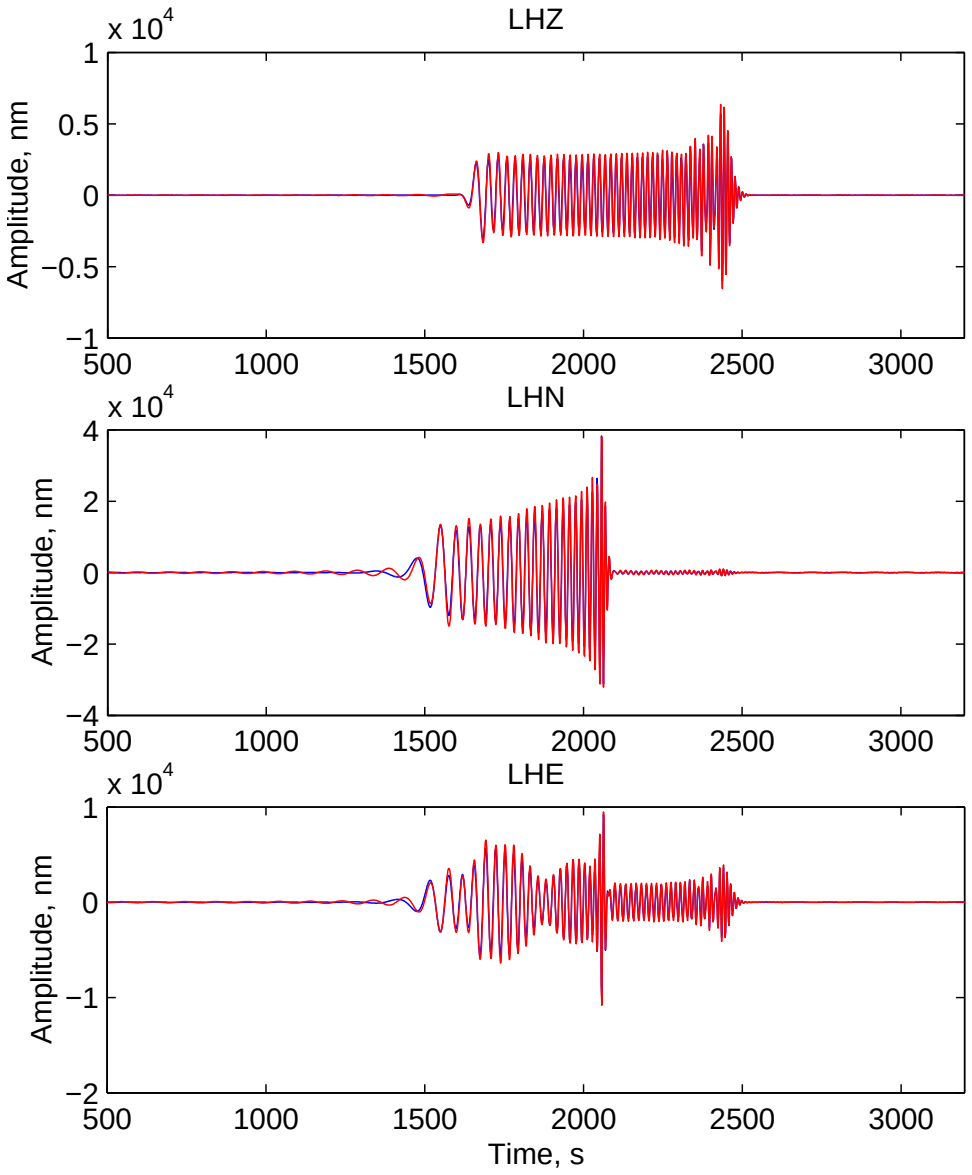


Figure B.3: Comparison with Herrmann's plane code, as in Figure B.1, but for station BILL.

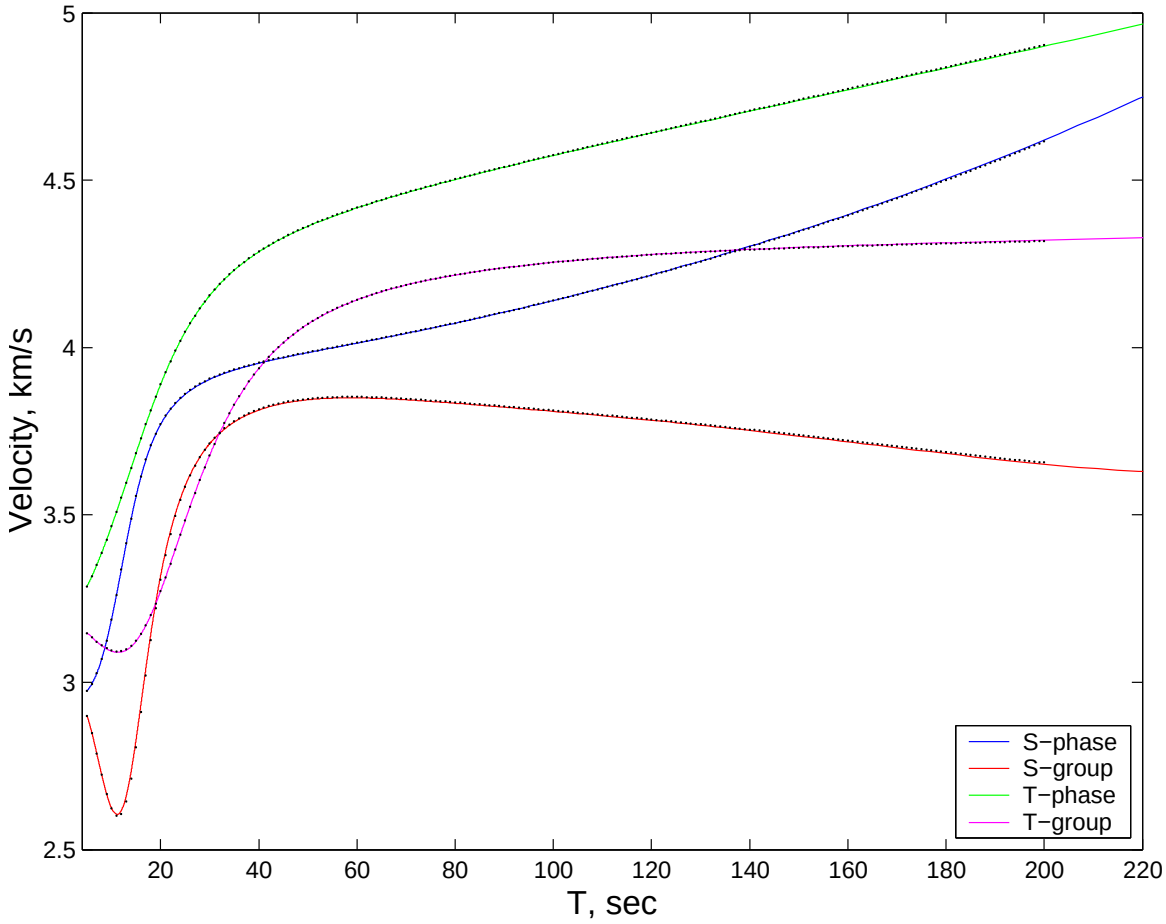


Figure B.4: Dispersion curves of phase and group velocities for spheroidal and toroidal fundamental modes. The solid color lines are the **Mineos** results, the (faint) black dotted lines are for the Herrmann's plane code. The solid line colors are blue for Rayleigh phase velocity, red for Rayleigh group velocity, green for Love phase velocity, and magenta for group Love velocity.

B.3 Mineos vs. SPECFEM3D_GLOBE

The **Mineos** synthetic seismograms were tested against SPECFEM3D_GLOBE synthetic seismograms for the same event and station set as described in the previous section.

B.3.1 Input 1D Model

The input model is an anisotropic, single-layered crust PREM with attenuation. The 3 km water layer is filled with crustal properties. SPECFEM3D_GLOBE has a special subroutine for evaluation of the model parameters by polynomial interpolation across a fixed number of layers from the center of the Earth up to the free surface at radius 6371 km. This polynomial representation was converted by a special program to a plain input file in **Mineos** format. The total number of vertical nodes is 237. The tabulated step by depth in the crust and upper mantle is close to 1 km.

B.3.2 SPECFEM3D_GLOBE Run Notes

SPECFEM3D_GLOBE was configured to make the synthetic seismogram (displacement, nm) 1 hour long. The Earth was split into 6 chunks. Each chunk consisted of 480×480 elements. So, the average lateral size of the elements near to the surface was 20×20 km. The state of some important run parameters were

- ELLIPTICITY - off
- TOPOGRAPHY - off
- ROTATION - off
- GRAVITY - on

As with **Mineos**, input coordinates are geographic, and geocentric coordinates are used internally.

B.3.3 Mineos Run Notes

Mineos was configured to compute all normal modes in the frequency range $0 - 0.2$ Hz and the radial mode range $0 \leq n \leq 400$. In total, the program computed 247565 spheroidal normal modes, 162154 toroidal modes, and 240 radial modes. Synthetic seismograms (acceleration, nm/s^2) 1 hour long were simulated. All seismograms were converted from acceleration to displacement in nm.

B.3.4 Tapering, Results Discussion

To reduce noise at spectral edges, all seismograms were half cosine tapered with corner frequencies (1/200, 1/100) Hz and (1/12, 1/10) Hz. Figures B.5 - B.16 illustrate three-component seismograms and amplitude spectra for the BJT, TLY, and BILL stations. SPECFEM3D_GLOBE results are plotted in red, **Mineos** in blue.

The test shows that synthetic seismograms and spectra for both methods are close. Attempts to increase the high-cut frequency, e.g., to 5 sec, led to differences in some places with periods close to 8 sec. This probably resulted because the SPECFEM3D_GLOBE spectral elements were not small enough.

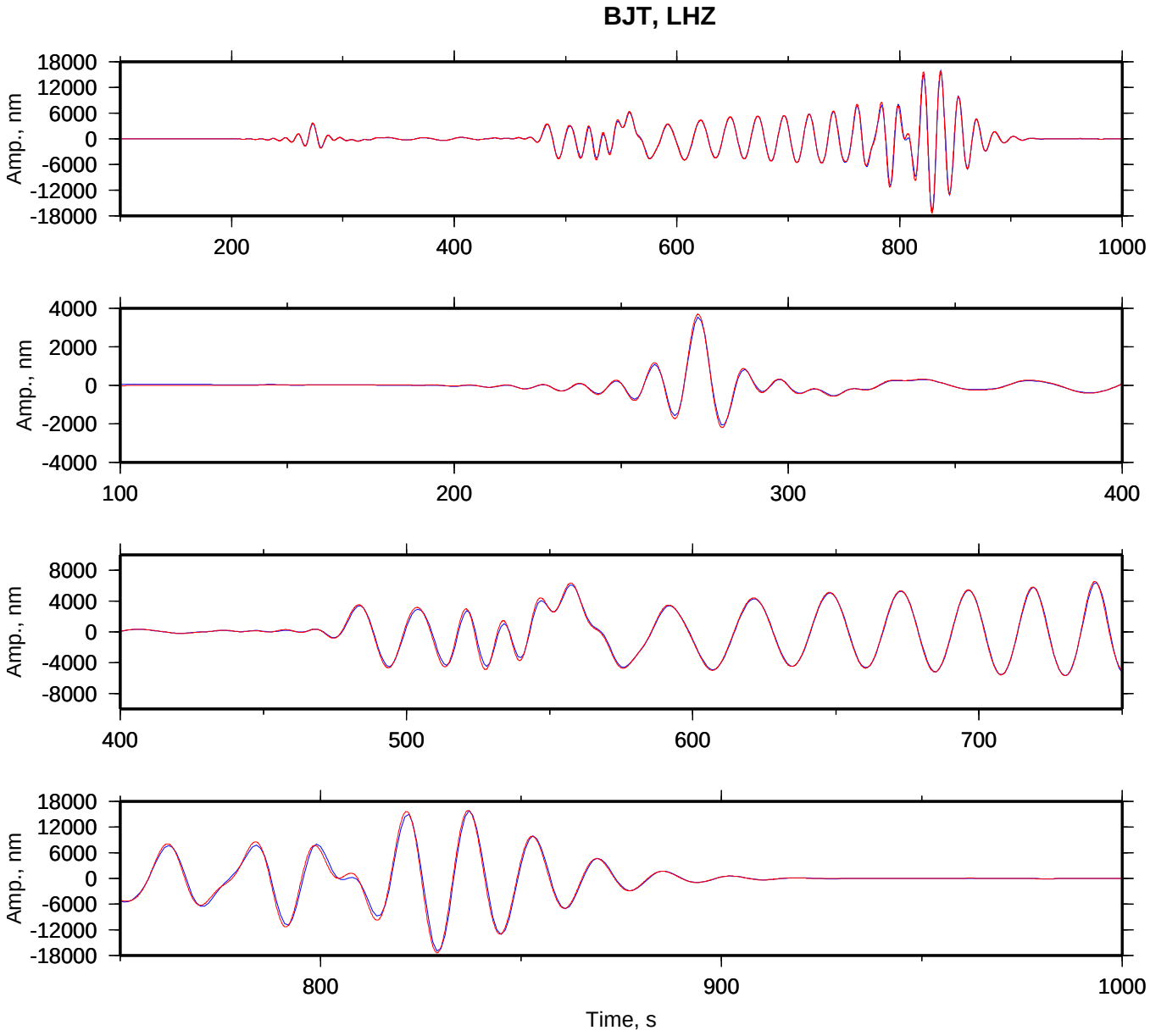


Figure B.5: Synthetic seismograms for SPECFEM3D_GLOBE (red) and **Mineos** (blue). Station BJT, channel LHZ. Distance = 19.123^o, Az = -135.267^o. The top plot shows the whole record; the others plot separate fragments.

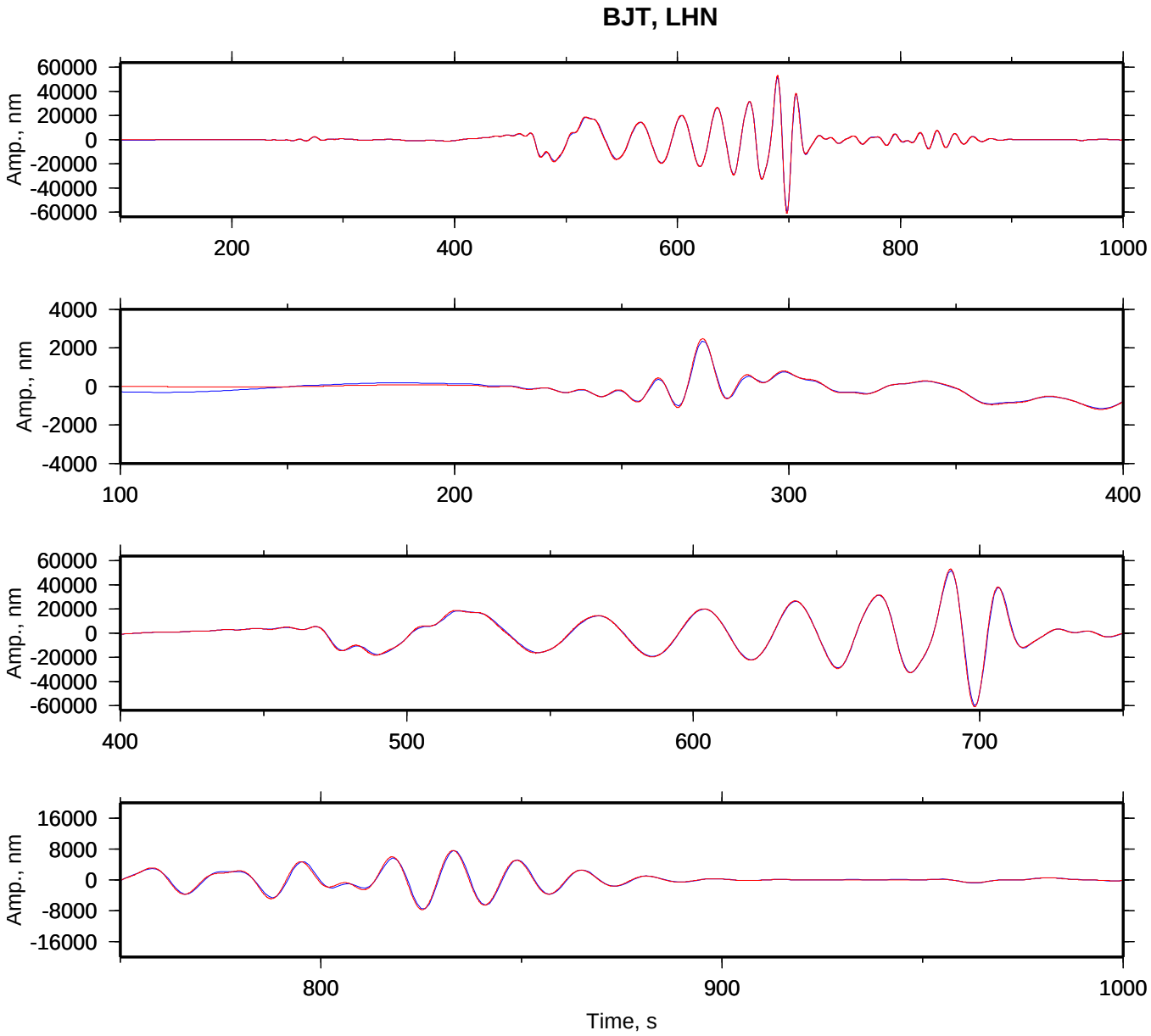


Figure B.6: The same as Figure B.5, but for the LHN channel.

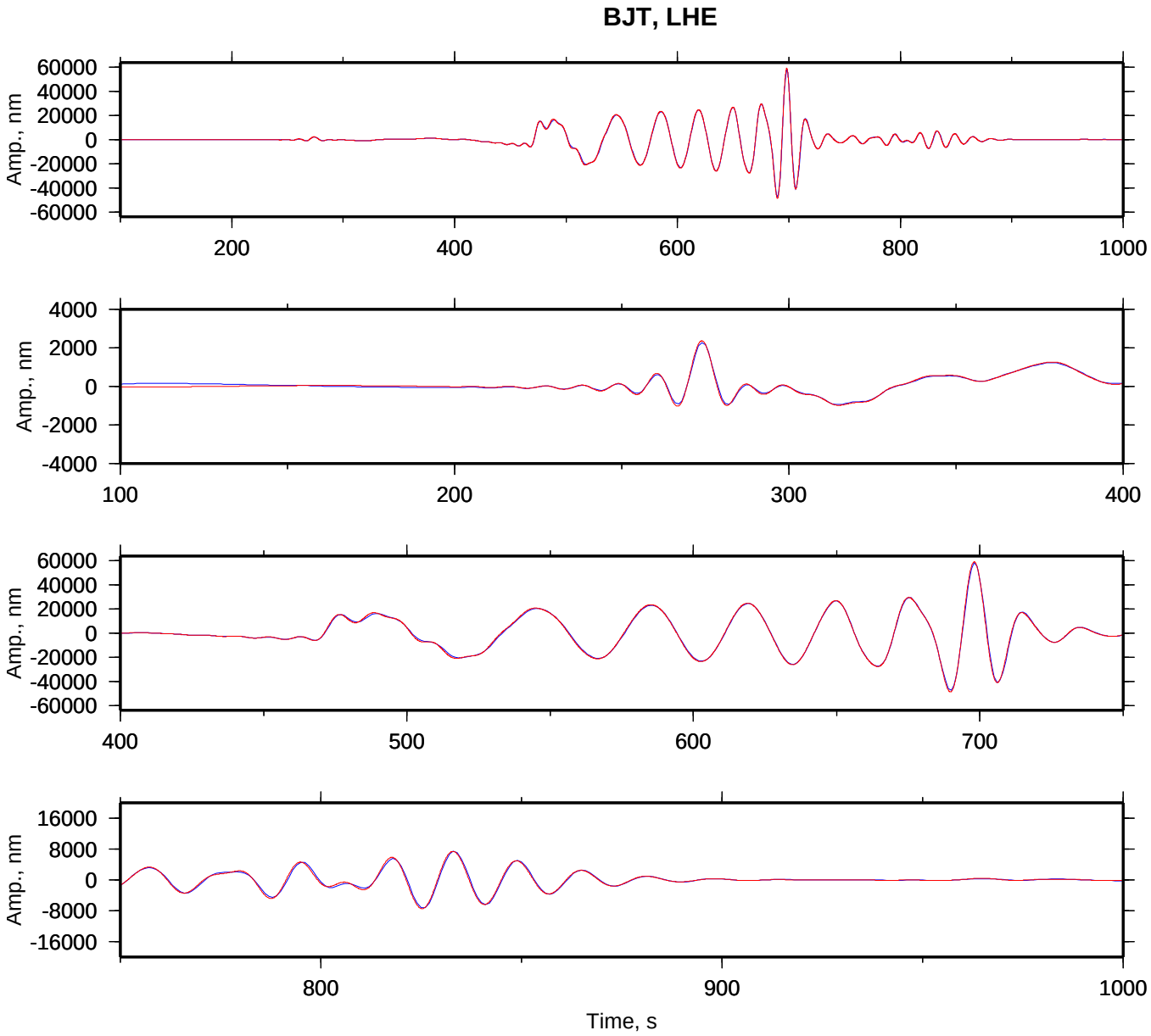


Figure B.7: The same as Figure B.5, but for the LHE channel.

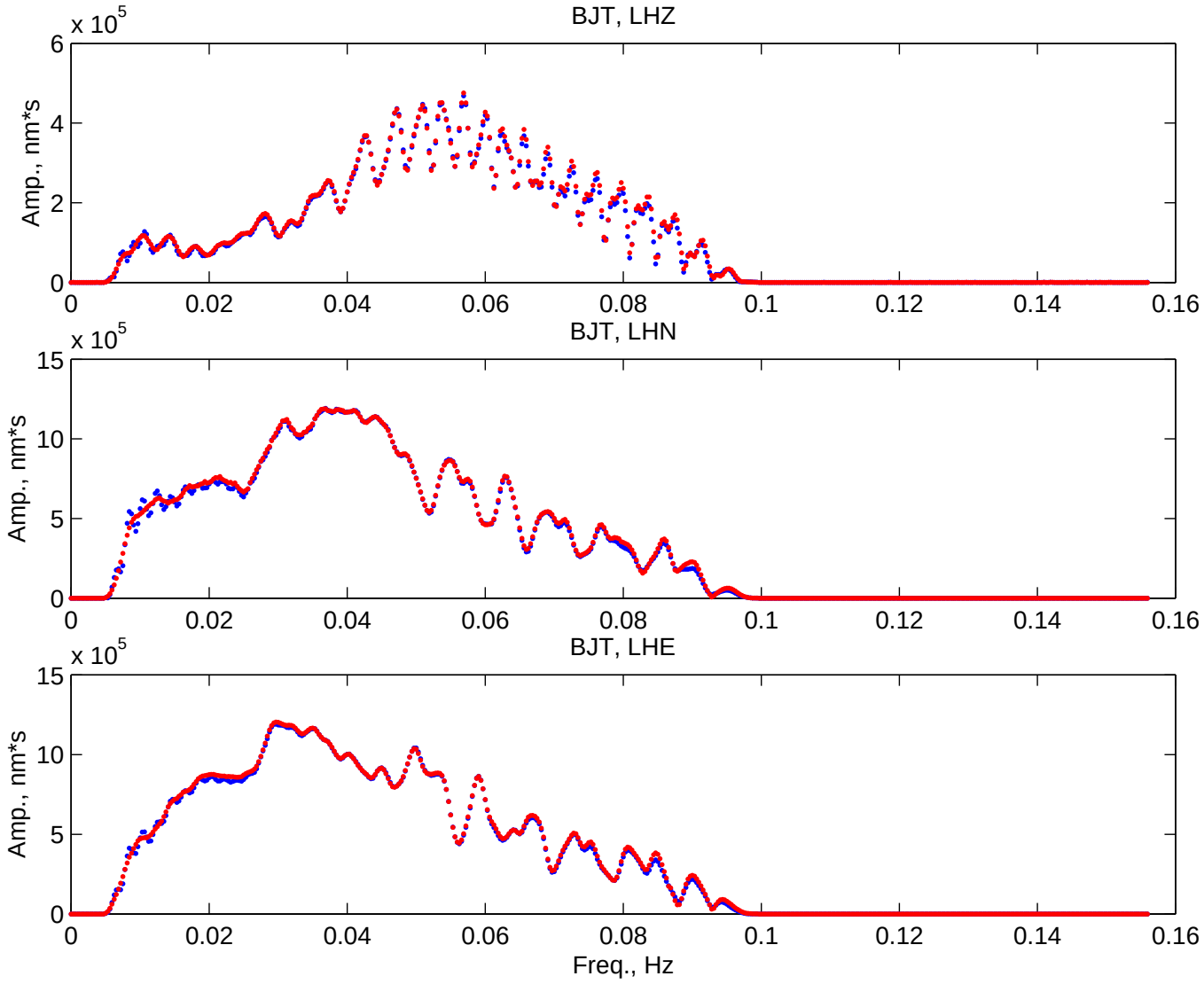


Figure B.8: Amplitude spectra for the station BJT. SPECFEM3D_GLOBE spectra (red), **Mineos** (blue).

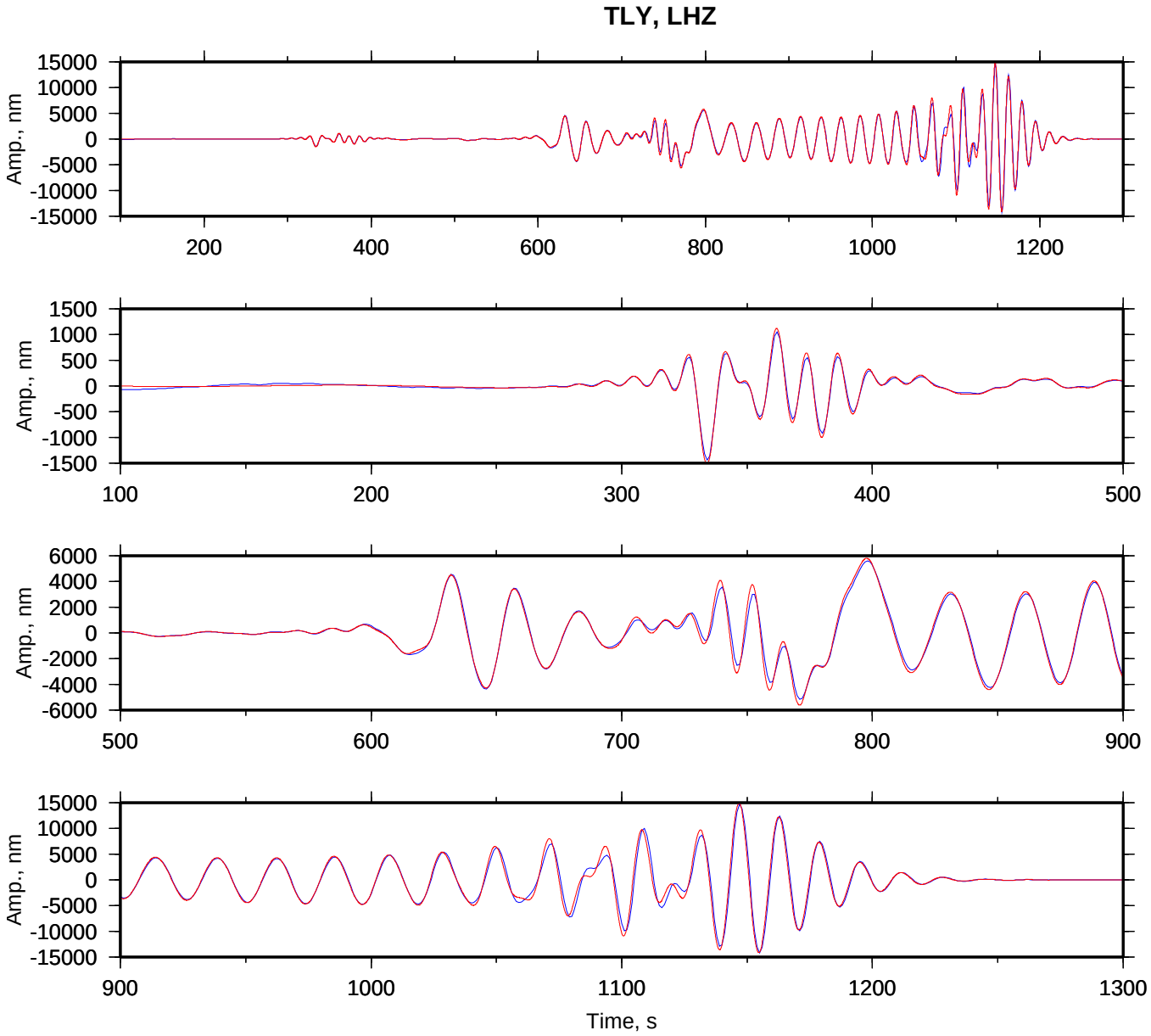


Figure B.9: Synthetic seismograms for SPECFEM3D_GLOBE (red) and **Mineos** (blue). Station TLY, channel LHZ. Distance = 26.308^o, Az = $-175,417^o$. The top plot shows the whole record; the others plot separate fragments.

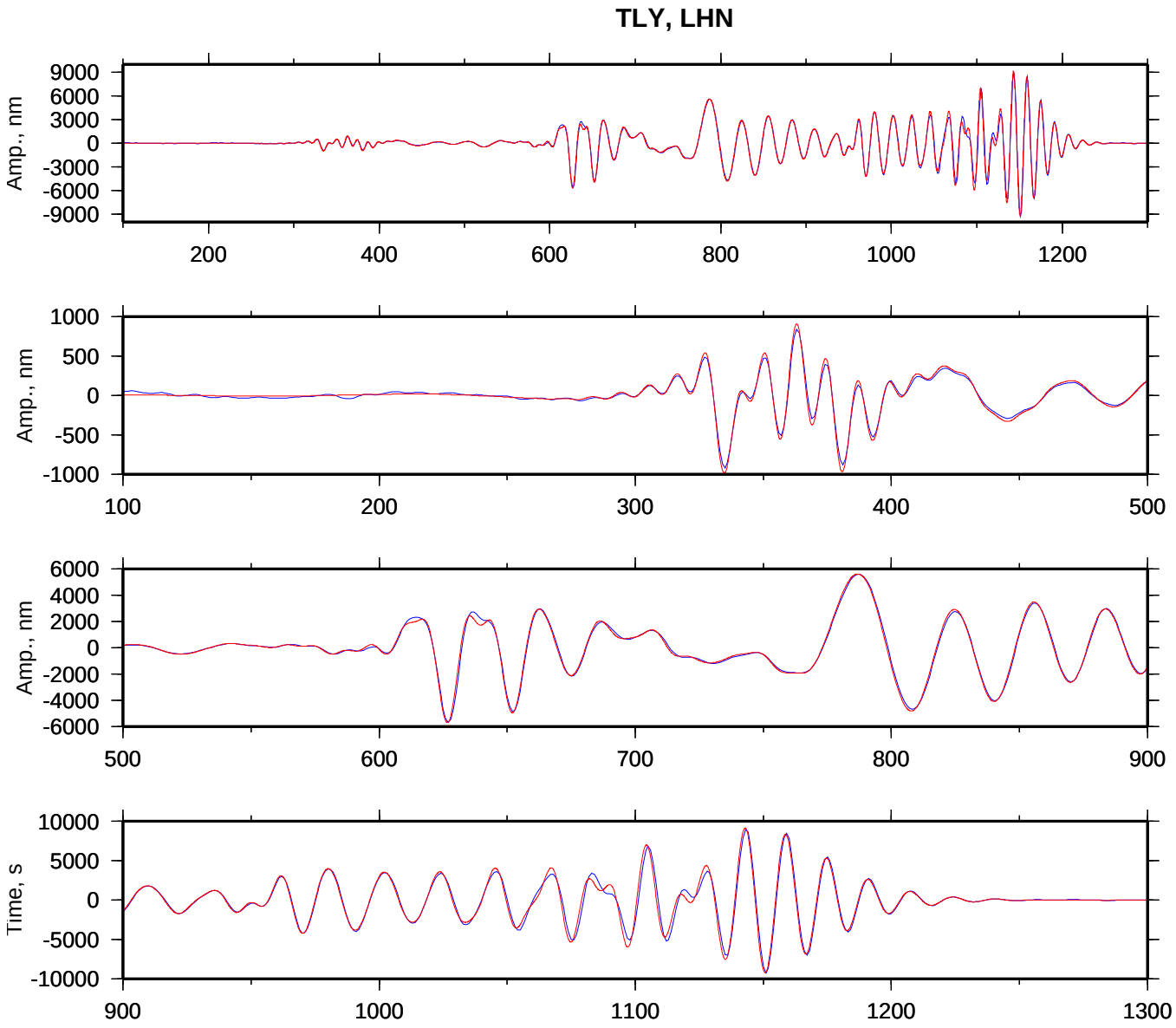


Figure B.10: The same as Figure B.9, but for the LHN channel.

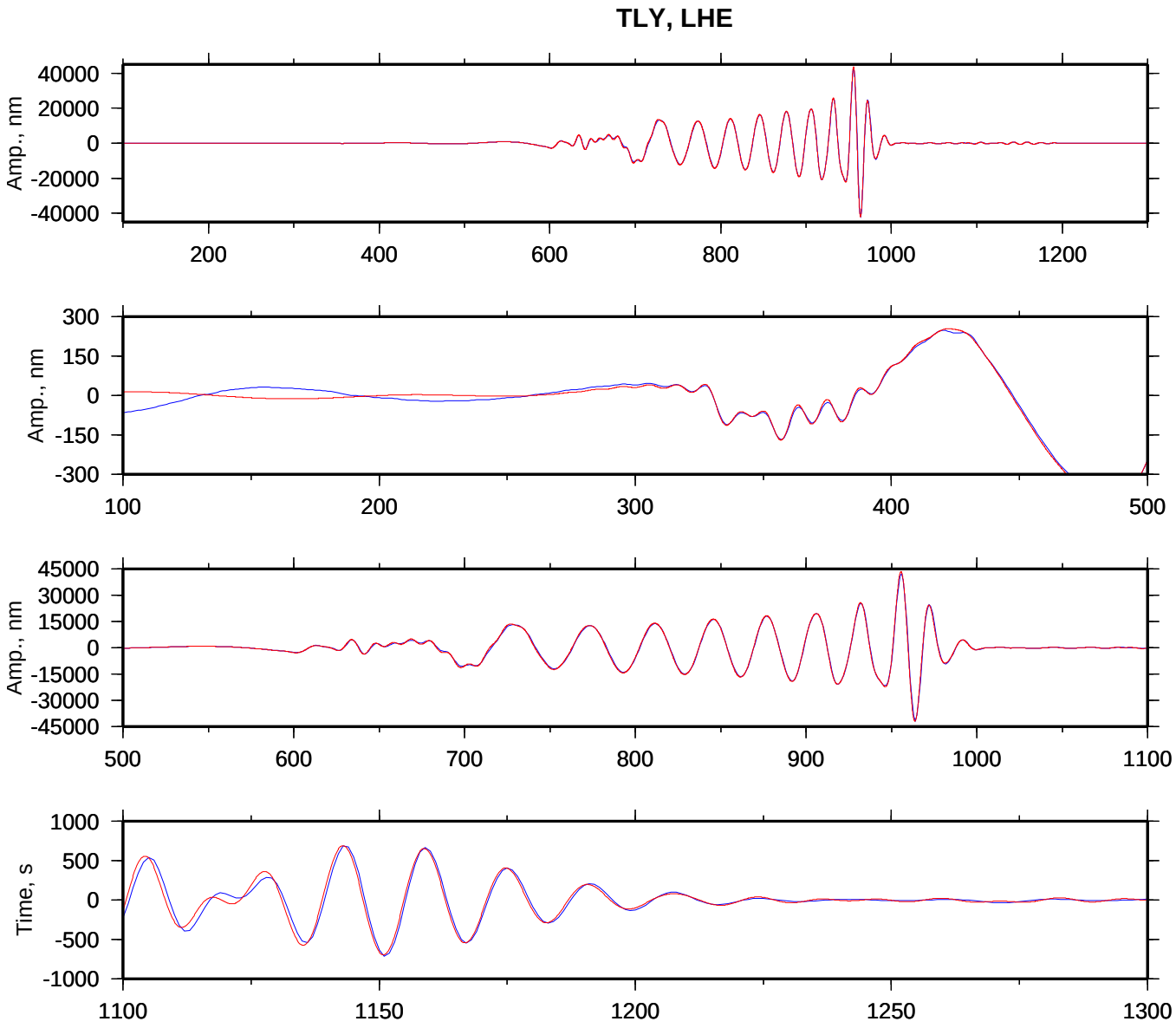


Figure B.11: The same as Figure B.9, but for the LHE channel.

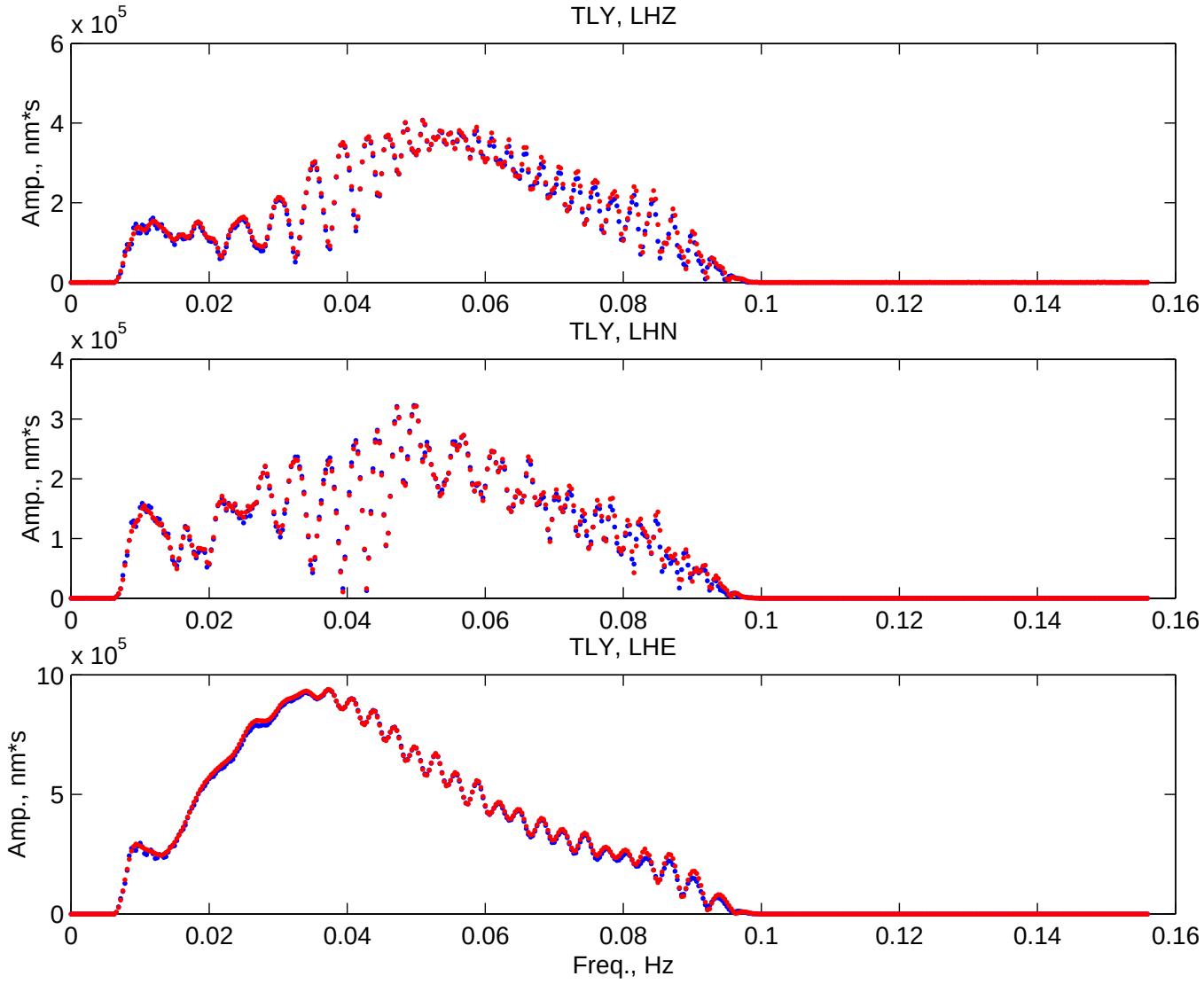


Figure B.12: Amplitude spectra for the station TLY. SPECFEM3D_GLOBE spectra (red), **Mineos** (blue).

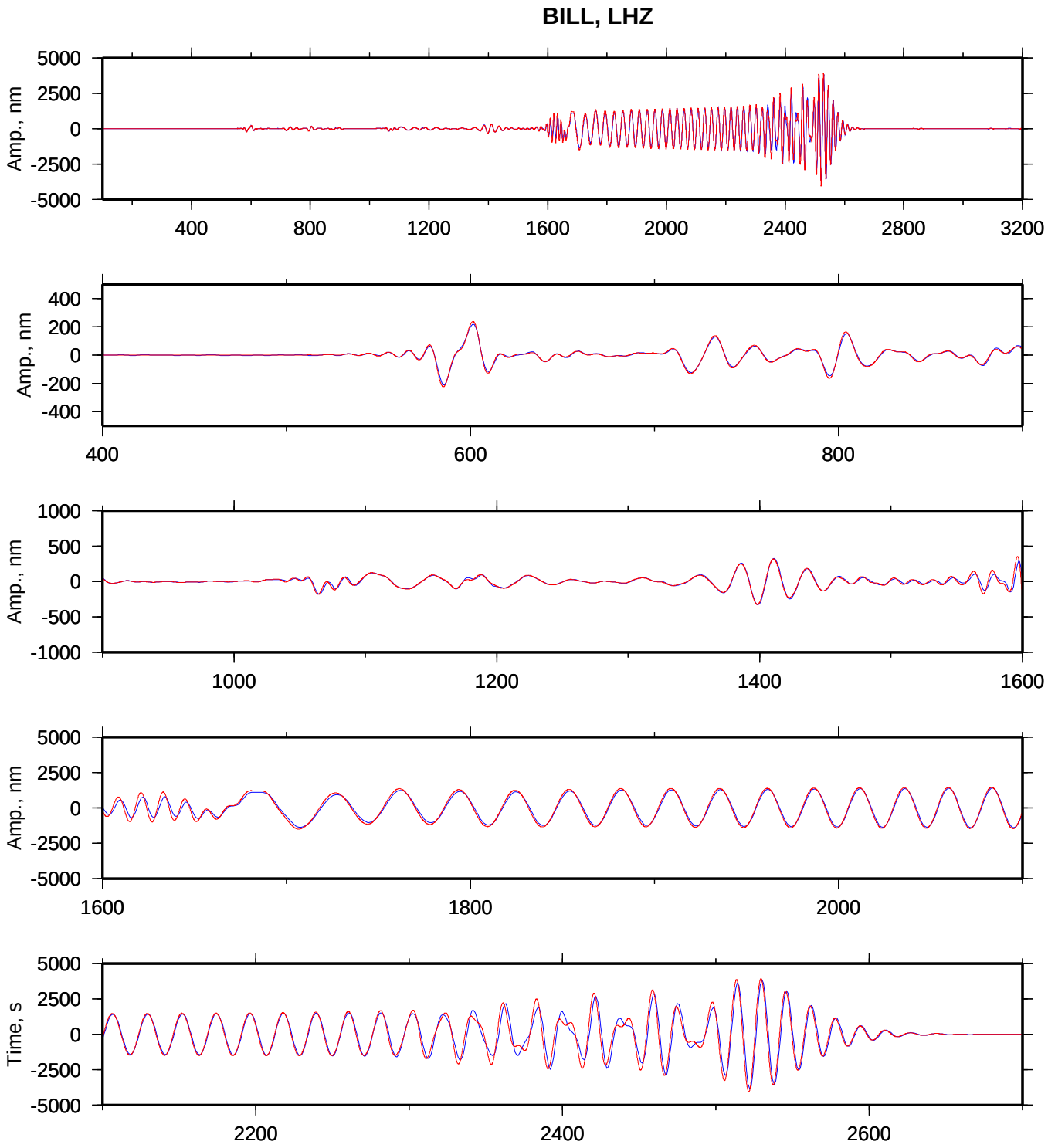


Figure B.13: Synthetic seismograms for SPECFEM3D_GLOBE (red) and **Mineos** (blue). Station BILL, channel LHZ. Distance = 57.417^o, Az = -103.266^o. The top plot shows the whole record; the others plot separate fragments.

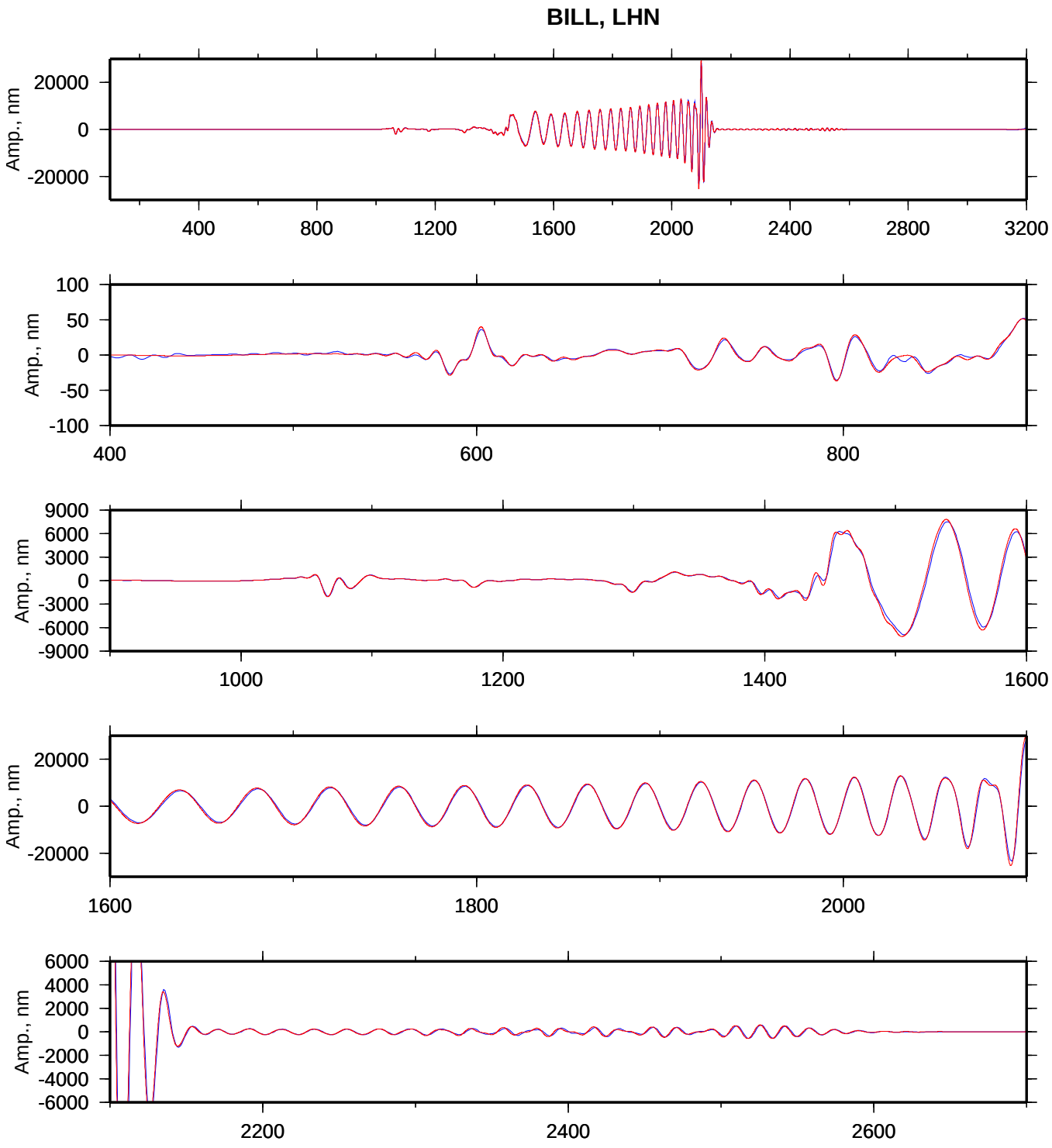


Figure B.14: The same as Figure B.13, but for the LHN channel.

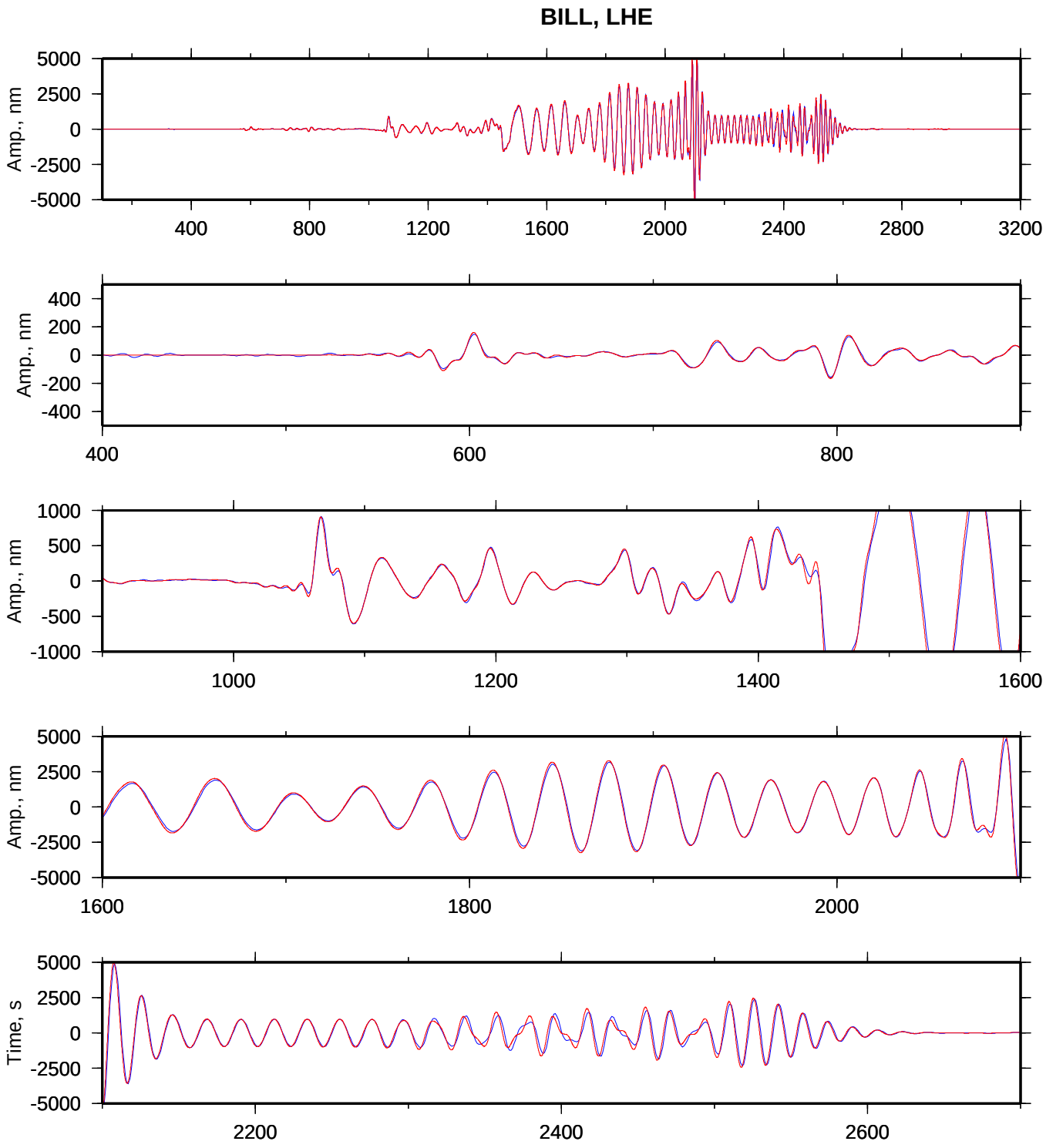


Figure B.15: The same as Figure B.13, but for the LHE channel.

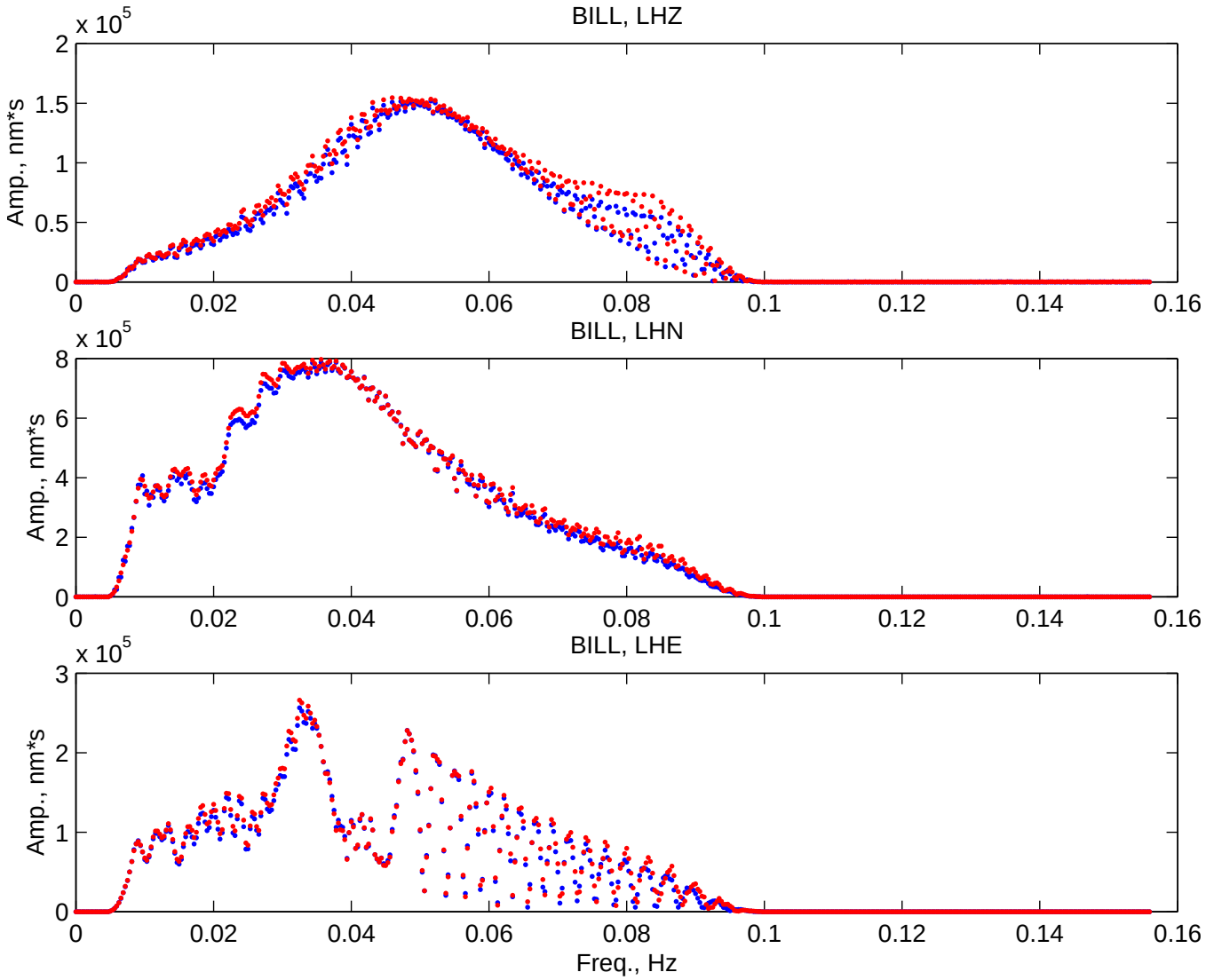


Figure B.16: Amplitude spectra for the station BILL. SPECFEM3D_GLOBE spectra (red), **Mineos** (blue).

Appendix C

Reference Frame Convention

The **Mineos** code uses the following convention for the Cartesian reference frame defining the standard sensor orientation:

- the x axis points East
- the y axis points North
- the z axis points up

Note that this convention is the same as for **SPECFEM3D_GLOBE** code, and it is different from the Harvard Centroid-Moment Tensor (CMT) convention. The Harvard CMT convention is

- the x axis points South
- the y axis points East
- the z axis points up

Appendix D

License

GNU GENERAL PUBLIC LICENSE Version 2, June 1991. Copyright (C) 1989, 1991 Free Software Foundation, Inc. 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software – to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps:

1. Copyright the software, and

2. Offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program" below refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification.") Each licensee is addressed as "you."

   Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

   You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

   (a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

   (b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

   (c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

   These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

   Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

   In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

   (a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

   (b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

   (c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

   The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

   If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

   Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version," you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found. For example:

> One line to give the program's name and a brief idea of what it does. Copyright © (year) (name of author)
>
> This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.
>
> This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.
>
> You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

> Gnomovision version 69, Copyright © year name of author Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type 'show w'. This is free software, and you are welcome to redistribute it under certain conditions; type 'show c' for details.

The hypothetical commands 'show w' and 'show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than 'show w' and 'show c'; they could even be mouse-clicks or menu items – whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

> Yoyodyne, Inc., hereby disclaims all copyright interest in the program 'Gnomovision' (which makes passes at compilers) written by James Hacker.
>
> (signature of Ty Coon)
> 1 April 1989
> Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.